```
LLL                 AAAAAAAAA     TTTTTTTTTTTTTTT
LLL                 AAAAAAAAA     TTTTTTTTTTTTTTT
LLL                 AAAAAAAAA     TTTTTTTTTTTTTTT
LLL              AAA         AAA        TTT
LLL              AAA         AAA        TTT
LLL              AAA         AAA        TTT
LLL              AAA         AAA        TTT
LLL              AAA         AAA        TTT
LLL              AAA         AAA        TTT
LLL              AAA         AAA        TTT
LLL              AAA         AAA        TTT
LLL              AAA         AAA        TTT
LLL              AAAAAAAAAAAAAAA        TTT
LLL              AAAAAAAAAAAAAAA        TTT
LLL              AAAAAAAAAAAAAAA        TTT
LLL              AAA         AAA        TTT
LLL              AAA         AAA        TTT
LLL              AAA         AAA        TTT
LLLLLLLLLLLLLLLL AAA         AAA        TTT
LLLLLLLLLLLLLLLL AAA         AAA        TTT
LLLLLLLLLLLLLLLL AAA         AAA        TTT
```

```
LL            AAAAAA    TTTTTTTTTT    CCCCCCCC  PPPPPPPP
LL            AAAAAA    TTTTTTTTTT    CCCCCCCC  PPPPPPPP
LL          AA      AA      TT      CC          PP      PP
LL          AA      AA      TT      CC          PP      PP
LL          AA      AA      TT      CC          PP      PP
LL          AA      AA      TT      CC          PP      PP
LL          AA      AA      TT      CC          PPPPPPPP
LL          AA      AA      TT      CC          PPPPPPPP
LL          AAAAAAAAAA      TT      CC          PP
LL          AAAAAAAAAA      TT      CC          PP
LL          AA      AA      TT      CC          PP            ....
LL          AA      AA      TT      CC          PP            ....
LLLLLLLLLL  AA      AA      TT        CCCCCCCC  PP            ....
LLLLLLLLLL  AA      AA      TT        CCCCCCCC  PP            ....


LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0 %TITLE  'LAT Control Program'
0 MODULE LATCP    (
0                 LANGUAGE (BLISS32),
0                 MAIN = LCP_MAIN,
0                 IDENT = 'V04-000'
0                 ) =
1 BEGIN
1
1 !
1 !*****************************************************************************
1 !*                                                                           *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
1 !*  ALL RIGHTS RESERVED.                                                     *
1 !*                                                                           *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
1 !*  TRANSFERRED.                                                             *
1 !*                                                                           *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
1 !*  CORPORATION.                                                             *
1 !*                                                                           *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
1 !*                                                                           *
1 !*                                                                           *
1 !*****************************************************************************
1 !
1
1 !++
1 ! FACILITY:     LAT Control Program
1 !
1 ! ABSTRACT:
1 !
1 !       This program controls and obtains counters and other information
1 !       from the LAT terminal port driver.
1 !
1 ! ENVIRONMENT:  VAX/VMS Operating System
1 !
1 ! AUTHOR:       Darrell Duffy   , CREATION DATE:  2-July-1983
1 !
1 ! Previously modified by:
1 !               Darrel Duffy
1 !               Joe Marchesani
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-009 RNG0009         Rod Gamache             16-Jul-1984
1 !               Use system NODEname for /NODE=NAME qualifier if SYS$NODE
1 !               does not translate.
1 !               Allow for the Physical UCB pointer to be zero in the check
1 !               for LT devices for shut requests.
1 !
```

```
0058  1 !        V03-008 RNG0008         Rod Gamache             7-Jun-1984
0059  1 !                Change IDENTIFICATION to SERVICE_IDENTIFICATION.
0060  1 !                With change in .CLD for /NAME and /IDENTIFICATION,
0061  1 !                the LATCP program must default values on it's own.
0062  1 !                Add HELP command.
0063  1 !                Add START HISTORY command.
0064  1 !                Add STOP HISTORY command.
0065  1 !                Add /[NO]LOG qualifier to START, SET and STOP verbs.
0066  1 !                Make proper determination of LAT terminals when shutting
0067  1 !                down.
0068  1 !
0069  1 !        V03-007 RNG0007         Rod Gamache             8-May-1984
0070  1 !                Change HOST qualifier to NODE qualifier, as per documentation
0071  1 !                change.
0072  1 !                Change MULTICAST_RATE to MULTICAST_TIMER.
0073  1 !
0074  1 !        V03-006 RNG0006         Rod Gamache             1-May-1984
0075  1 !                Change LATCP to be a utility, instead of DCL command.
0076  1 !                                                                        53
0077  1 !        V03-005 LMP0221         L. Mark Pilant,         12-Apr-1984  14:42
0078  1 !                Change UCB$L_OWNUIC to ORB$L_OWNER and UCB$W_VPROT to
0079  1 !                ORB$W_PROT.
0080  1 !                                                                        52
0081  1 !        V03-004 RNG0004                 Rod Gamache             27-Mar-1984
0082  1 !                Add SHOW COUNTERS/DEVICE, SHOW COUNTERS/SERVERS and
0083  1 !                SHOW SERVERS commands.
0084  1 !                Add START/SET NODE_ID command.
0085  1 !
0086  1 !        V03-003 RNG0003                 Rod Gamache             14-Mar-1984
0087  1 !                Fix references to NMADEF.
0088  1 !
0089  1 !        V03-002 RNG0002                 Rod Gamache              2-Mar-1984
0090  1 !                Add circuit counters.
0091  1 !
0092  1 !        V03-001 RNG0001                 Rod Gamache             31-Jan-1984
0093  1 !                Add new V5 message formats.
0094  1 !--
```

41

00

6E
20

20
21

```
%SBTTL  'Definitions'

!
! INCLUDE FILES:
!

LIBRARY 'SYS$LIBRARY:LIB';
LIBRARY 'SHRLIB$:NMALIBRY';
REQUIRE 'OBJ$:LATDEF';

!
! LINKAGES
!

    LINKAGE
        ALONONPAGED = JSB (REGISTER = 1; REGISTER = 2),
        DEANONPAGED = JSB (REGISTER = 0)
                : NOPRESERVE (1,2,3),
        VERIFYCHAN = JSB (REGISTER = 0; REGISTER = 1)
                : NOPRESERVE (2,3,4,5),
        STARTSTOP = JSB ( REGISTER = 5 )
                : NOPRESERVE (2,3,4,5,6,7,8,9,10,11)
        ;

!
! TABLE OF CONTENTS:
!

FORWARD ROUTINE
    LCP_MAIN,                           ! Main entry
    LCP_QUAL,                           ! Convert all qualifiers
    LCP_GETSTR,                         ! Obtain a string qualifier
    LCP_FIXNAME,                        ! Fixup a name string qualifier
    LCPK_FINDDRIVER,                    ! Is the port driver loaded?
    LCP_START : NOVALUE,                ! Start the driver
    LCP_STOP : NOVALUE,                 ! Stop the driver
    LCP_SET,                            ! Set the multicast message
    LCP_SHOW : NOVALUE,                 ! Process a SHOW command
    LCP_EXIT : NOVALUE,                 ! Exit LCP command
    LCP_ZERO : NOVALUE,                 ! Zero the counters
    LCPK_SET,                           ! Kernel mode set multicast routine
    LCPK_START,                         ! Kernel mode start routine
    LCPK_STOP,                          ! Kernel mode stop routine
    LCPK_ZERO : NOVALUE,                ! Kernel mode zero routine
    LCPK_STARTHISTORY,                  ! Kernel mode start history routine
    LCPK_STOPHISTORY : NOVALUE,         ! Kernel mode stop history routine
    LCPK_IS_LAT_TERMINAL,               ! Kernel mode find LAT terminal routine
    LCP_SHOWCOUQUAL : NOVALUE,          ! Show counters qualifiers
    LCP_SHOWCOU : NOVALUE,              ! Show counters
    LCP_SHOWNODECOU : NOVALUE,          ! Show node counters
    LCP_SHOWDEVCOU :NOVALUE,            ! Show Ethernet device counters
    LCPK_SHOWDEVCOU,                    ! Show Ethernet device counters
    LCP_DISPDEVCOU,                     ! Display the Ethernet device ctrs
    LCP_FINDCOU,                        ! Find the counter value
    LCP_SHOWSERVCOU : NOVALUE,          ! Show LAT Server counters
    LCP_SHOWHIST : NOVALUE,             ! Show the history
    LCP_SHOWCHAR : NOVALUE,             ! Show the characteristics
```

```
153          1     LCP_SHOWSERVERS : NOVALUE,          ! Show the LAT Servers                                          64
154          1     LCP_HELP : NOVALUE,                 ! Give HELP
155          1     LCP_CVTGROUPS : NOVALUE,            ! Convert group code  to an ascii string                        2F
156          1     LCP_FAOOUTPUT : NOVALUE,            ! Convert and output a string                                   61
157          1     LCPK_GETCOUNTERS,                   ! Obtain counters from driver
158          1     LCPK_GETSERVCTRS,                   ! Obtain CSB counters from driver                               20
159          1     LCPK_GETSERVERS,                    ! Obtain CSB server information                                 6D
160          1     LCPK_GETHISTORY,                    ! Obtain history buffer from driver
161          1     LCPK_GETCOMMON                      ! Obtain common buffer area from driver                         6F
162          1     ;                                                                                                   6C
163          1
164          1 !                                                                                                       64
165          1 ! MACROS:                                                                                               73
166          1 !
167          1                                                                                                         63
168          1 MACRO                                                                                                   73
169          1 !
170          1 ! Set up a static descriptor                                                                            65
171          1 !                                                                                                       6C
172        M 1     DESCRIPTOR (STRING) =
173          1         UPLIT (%CHARCOUNT (STRING), UPLIT BYTE (STRING))%,                                              6E
174          1 !                                                                                                       6C
175          1 ! Set up a dynamic descriptor
176          1 !                                                                                                       73
177        M 1     $SETDSC (DSC, SIZE, ADR) =                                                                          20
178        M 1         (
179        M 1         DSC [DSC$B_CLASS] = DSC [DSC$B_DTYPE] = 0;                                                      69
180        M 1         DSC [DSC$W_LENGTH] = SIZE;                                                                      74
181        M 1         DSC [DSC$A_POINTER] = ADR
182          1         )%,                                                                                             20
183          1 !                                                                                                       21
184          1 ! Compare a pair of descriptors
185          1 !                                                                                                       72
186        M 1     $CMPDSC (DSC1, DSC2) =                                                                              79
187        M 1         (
188        M 1         CH$EQL                                                                                          2F
189        M 1             (                                                                                           75
190        M 1             .DSC1 [DSC$W_LENGTH],
191        M 1             .DSC1 [DSC$A_POINTER],
192        M 1             MAX ( MIN ( .DSC1 [DSC$W_LENGTH], .DSC2 [DSC$W_LENGTH]), 3),
193        M 1             .DSC2 [DSC$A_POINTER],
194        M 1             )
195        M 1             )
196          1         )%,
197          1 !
198          1 !  LOG information if requested
199          1 !
200        M 1     LOG_SIGNAL [] =
201        M 1         (
202        M 1         IF (CLI$PRESENT (QUAL_LOG) )
203        M 1         THEN
204        M 1             SIGNAL (%REMAINING)
205        M 1             ;
206          1         )%,
207          1
208        M 1     LOG_OUTPUT [] =
209        M 1         (
```

```
M 0346  1          IF (CLI$PRESENT (QUAL_LOG) )
M 0347  1          THEN
M 0348  1              LIB$PUT_OUTPUT (%REMAINING)
M 0349  1          ;
   0350  1          }%
   0351  1      ;
   0352  1
   0353  1      !
   0354  1      !  Define FLAGS bits
   0355  1      !
   0356  1      MACRO
   0357  1          FLAG_V_SERVERS  = 0,0,1,0%,
   0358  1          FLAG_V_DEVICE   = 0,1,1,0%,
   0359  1          FLAG_V_NODE     = 0,2,1,0%,
   0360  1
   0361  1          FLAG_L_ALL      = 0,0,32,0%
   0362  1      ;
   0363  1
   0364  1  !
   0365  1  ! EQUATED SYMBOLS:
   0366  1  !
   0367  1
   0368  1  LITERAL
   0369  1      TRUE = 1,
   0370  1      FALSE = 0,
   0371  1      SUCCESS = 1,
   0372  1      FAILURE = 0,
   0373  1
   0374  1      LF = 10,                          ! Line Feed character
   0375  1      CR = 13,                          ! Carriage return character
   0376  1
   0377  1      INPUT_LEN = 1024,                 ! Input command line length
   0378  1      CMDLEN = 16,                      ! Subcommand string length
   0379  1      NUMLEN = 16                       ! Numbers
   0380  1      ;
   0381  1
   0382  1  BIND
   0383  1      LCP_PROMPT          = %ASCID 'LCP> ',
   0384  1      QUAL_SUBCOMMAND     = %ASCID 'SUBCOMMAND',
   0385  1      QUAL_NAME           = %ASCID 'NODE_NAME' : BLOCK [,BYTE],
   0386  1      QUAL_ID             = %ASCID 'SERVICE_IDENTIFICATION' : BLOCK [,BYTE],
   0387  1      QUAL_GROUPS         = %ASCID 'GROUPS' : BLOCK [,BYTE],
   0388  1      QUAL_RATING         = %ASCID 'RATING' : BLOCK [,BYTE],
   0389  1      QUAL_SERV_RATING    = %ASCID 'RATING_SERVICE' : BLOCK [,BYTE],
   0390  1      QUAL_SERV_NAME      = %ASCID 'SERVICE_NAME' : BLOCK [,BYTE],
   0391  1      QUAL_START          = %ASCID 'START' : BLOCK [,BYTE],
   0392  1      QUAL_STOP           = %ASCID 'STOP' : BLOCK [,BYTE],
   0393  1      QUAL_SHOW           = %ASCID 'SHOW' : BLOCK [,BYTE],
   0394  1      QUAL_SET            = %ASCID 'SET ' : BLOCK [,BYTE],
   0395  1      QUAL_ZERO           = %ASCID 'ZERO' : BLOCK [,BYTE],
   0396  1      QUAL_COUNTERS       = %ASCID 'COUNTERS' : BLOCK [,BYTE],
   0397  1      QUAL_HISTORY        = %ASCID 'HISTORY' : BLOCK [,BYTE],
   0398  1      QUAL_CHAR           = %ASCID 'CHARACTERISTICS' : BLOCK [,BYTE],
   0399  1      QUAL_MCAST          = %ASCID 'MULTICAST_TIMER'  : BLOCK [,BYTE],
   0400  1      QUAL_SERVERS        = %ASCID 'SERVERS' : BLOCK [,BYTE],
   0401  1      QUAL_DEVICE         = %ASCID 'DEVICE' : BLOCK [,BYTE],
   0402  1      QUAL_NODE           = %ASCID 'NODE' : BLOCK [,BYTE],
```

```
  0403  1      QUAL_LOG            = %ASCID 'LOG'  : BLOCK [,BYTE],
  0404  1      QUAL_NODEID         = %ASCID 'NODE_IDENTIFICATION' : BLOCK [,BYTE],
  0405  1      QUAL_NONE           = %ASCID 'none available' : BLOCK [,BYTE]
  0406  1      ;
  0407  1
  0408  1 BIND
  0409  1      ACTIVE = %ASCID 'active',
  0410  1      INACTIVE = %ASCID 'inactive'
  0411  1      ;
  0412  1
  0413  1
  0414  1      BIND
L 0415  1          COUNTERSCTRSTR = %ASCID %STRING
L 0416  1              (                                                        61
L 0417  1              '!/',                                                    65
L 0418  1              'LCP Node Counters!/!/',
L 0419  1              '!10UL   Receive frames!/',
L 0420  1              '!10UL   Receive errors!/',
L 0421  1              '!10UL   Receive duplicates!/',
L 0422  1              '!10UL   Transmit frames!/',                             63
L 0423  1              '!10UL   Transmit errors!/',                             65
L 0424  1              '!10XL   Last transmit failure code!/',
L 0425  1              '!10UL   Retransmissions!/',
L 0426  1              '!10UL   Circuit timeouts!/',
L 0427  1              '!10UL   Protocol errors!/',                             69
L 0428  1              '!10XL   Protocol bit mask!/',                           20
L 0429  1              '!10UL   Resource errors!/',
L 0430  1              '!10UL   No transmit buffer!/',
L 0431  1              '!10UL   Unit timeouts!/',
  0432  1              );                                                       69
  0433  1                                                                       61
  0434  1      BIND
L 0435  1          PROTOCOLMASK = %ASCID %STRING
L 0436  1              (
L 0437  1              %CHAR (CR, LF),
L 0438  1              ' Protocol errors include:',                             69
L 0439  1              %CHAR (CR, LF)                                           67
  0440  1              );
  0441  1
  0442  1
  0443  1      BIND
L 0444  1          SERVCTRSTR = %ASCID %STRING                                  71
L 0445  1              (                                                        65
L 0446  1              '!/',
L 0447  1              'LCP Server Counters for !AS!/!/',                       00
L 0448  1              '!10UL   Receive frames!/',
L 0449  1              '!10UL   Transmit frames!/',
L 0450  1              '!10UL   Retransmissions!/',
L 0451  1              '!10UL   Out of sequence frames!/',                      73
L 0452  1              '!10UL   Invalid messages!/',                            74
L 0453  1              '!10UL   Invalid slots!/',
L 0454  1              '!/'
  0455  1              );
  0456  1                                                                       72
  0457  1                                                                       00
  0458  1 !
  0459  1 !  NOTE: The following 2 tables must be kept in order!
```

```
  324          0460  1 !        If the counters are to be displayed correctly.
  325          0461  1     BIND
  326        L 0462  1         DEVCTRSTR = %ASCID %STRING
  327        L 0463  1             (
  328        L 0464  1             '!/',
  329        L 0465  1             'LCP Ethernet Device Counters!/!/',
  330        L 0466  1             '!10UL   Seconds since last zeroed!/',
  331        L 0467  1             '!10UL   Receive frames!/',
  332        L 0468  1             '!10UL   Receive errors!/',
  333        L 0469  1             '!10UL   Multicast frames received!/',
  334        L 0470  1             '!10UL   Receive bytes!/',
  335        L 0471  1             '!10UL   Multicast bytes received!/',
  336        L 0472  1             '!10UL   Transmit frames!/',
  337        L 0473  1             '!10UL   Transmit errors!/',
  338        L 0474  1             '!10UL   Multicast frames transmitted:/',
  339        L 0475  1             '!10UL   Transmit bytes!/',
  340        L 0476  1             '!10UL   Multicast bytes transmitted!/',
  341        L 0477  1             '!10UL   Frames sent, single collision!/',
  342        L 0478  1             '!10UL   Frames sent, multiple collisions!/',
  343        L 0479  1             '!10UL   Frames sent, initially deferred!/',
  344        L 0480  1             '!10UL   Transmit collision detect check failure!/',
  345        L 0481  1             '!10UL   Data overrun!/',
  346        L 0482  1             '!10UL   Local buffer errors!/',
  347        L 0483  1             '!10UL   System buffer unavailable!/',
  348        L 0484  1             '!10UL   User buffer unavailable!/',
  349          0485  1             );
  350          0486  1
  351          0487  1 BIND
  352          0488  1     CTRNAMES = UPLIT
  353          0489  1         (
  354          0490  1         LONG    (NMA$C_CTLIN_ZER),
  355          0491  1         LONG    (NMA$C_CTLIN_DBR),
  356          0492  1         LONG    (NMA$C_CTLIN_RFL),
  357          0493  1         LONG    (NMA$C_CTLIN_MBL),
  358          0494  1         LONG    (NMA$C_CTLIN_BRC),
  359          0495  1         LONG    (NMA$C_CTLIN_MBY),
  360          0496  1         LONG    (NMA$C_CTLIN_DBS),
  361          0497  1         LONG    (NMA$C_CTLIN_SFL),
  362          0498  1         LONG    (NMA$C_CTLIN_MBS),
  363          0499  1         LONG    (NMA$C_CTLIN_BSN),
  364          0500  1         LONG    (NMA$C_CTLIN_MSN),
  365          0501  1         LONG    (NMA$C_CTLIN_BS1),
  366          0502  1         LONG    (NMA$C_CTLIN_BSM),
  367          0503  1         LONG    (NMA$C_CTLIN_BID),
  368          0504  1         LONG    (NMA$C_CTLIN_CDC),
  369          0505  1         LONG    (NMA$C_CTLIN_OVR),
  370          0506  1         LONG    (NMA$C_CTLIN_LBE),
  371          0507  1         LONG    (NMA$C_CTLIN_SBU),
  372          0508  1         LONG    (NMA$C_CTLIN_UBU),
  373          0509  1         -1
  374          0510  1         );
  375          0511  1
  376          0512  1     BIND
  377        L 0513  1         RCVERRMASK = %ASCID %STRING
  378        L 0514  1             (
  379        L 0515  1             %CHAR (CR, LF),
  380        L 0516  1             ' Receive errors include:',
```

```
L 0517  1             %CHAR (CR, LF)
  0518  1             );
  0519  1
  0520  1     BIND
L 0521  1         XMTERRMASK = %ASCID %STRING
L 0522  1             (
L 0523  1             %CHAR (CR, LF),
L 0524  1             ' Transmit errors include:',
L 0525  1             %CHAR (CR, LF)
  0526  1             );
  0527  1
  0528  1 BIND
L 0529  1     faostr1 = %ascid %string
L 0530  1         (
L 0531  1         '!/',
L 0532  1         '     Time       Type Slots Idx Seq   Idx Seq   Seq Ack    DATA...',
L 0533  1         '!/'
  0534  1         ),
L 0535  1     faostr2 = %ascid %string
L 0536  1         (
L 0537  1         ' !14<!%T!>  !XB   !XB    !XB  !XB    !XB  !XB    !XB !XB     !XB !XB',
L 0538  1         '  !XB !XB  !XB !XB  !XB !XB  !XB !XB  !XB !XB  !XB !XB  !XB !XB !/'
  0539  1         );
  0540  1
  0541  1 !
  0542  1 ! List of device names to try in order.  A logical name allows us to
  0543  1 ! define a new datalink device to use or to use a controller other than
  0544  1 ! the first.
  0545  1 ! We assume that the UNA is first, and we allow the QNA driver to be
  0546  1 ! used also.
  0547  1 !
  0548  1 BIND
  0549  1     DEVNAMES = UPLIT
  0550  1         (
  0551  1         %ASCID 'LAT$DEVICE',
  0552  1         %ASCID 'XEA0:',
  0553  1         %ASCID 'XQA0:',
  0554  1         0
  0555  1         );
  0556  1
  0557  1 !
  0558  1 ! The list of parameters for the datalink driver to start up.  We include
  0559  1 ! the protocol type here as well as the other parameters to allow the device
  0560  1 ! to be started with or without DECnet.
  0561  1 !
  0562  1
  0563  1 BIND
  0564  1     SETPARM = UPLIT
  0565  1         (
  0566  1         WORD    (NMA$C_PCLI_BUS),       ! Buffer size
  0567  1         LONG    (1500),
  0568  1         WORD    (NMA$C_PCLI_BFN),       ! Number of buffers
  0569  1         LONG    (1),
  0570  1         WORD    (NMA$C_PCLI_PAD),       ! Pad short buffers (NO!)
  0571  1         LONG    (NMA$C_STATE_OFF),
  0572  1         WORD    (NMA$C_PCLI_PTY),       ! Protocol type
  0573  1         LONG    (%X'0480'),             ! Digital assigned protcol type
```

```
0574  1          WORD    (NMA$C_PCLI_PHA),        ! Physical address
0575  1          WORD    (2),                     ! Length of string
0576  1          WORD    (NMA$C_LINMC_SDF),       ! Use default setting
0577  1          WORD    (NMA$C_PCLI_PRM),        ! Promiscous mode
0578  1          LONG    (NMA$C_STATE_OFF),
0579  1          WORD    (NMA$C_PCLI_DCH),        ! Data chaining
0580  1          LONG    (NMA$C_STATE_OFF),
0581  1          WORD    (NMA$C_PCLI_CRC),        ! Generate crc on transmit
0582  1          LONG    (NMA$C_STATE_ON),
0583  1          WORD    (NMA$C_PCLI_CON),        ! Controller mode
0584  1          LONG    (NMA$C_LINCN_NOR)        ! Normal
0585  1          ),
0586  1      SETPARMEND = UPLIT BYTE(0),
0587  1      SETPARMDSC = UPLIT (SETPARMEND - SETPARM, SETPARM)
0588  1      ;
0589  1
0590  1 !
0591  1 ! Build a list of Global Host Node protocol error bit values and their names
0592  1 !
0593  1
M 0594  1 MACRO GHBERRBITS [BITNAME, MEANING] =
M 0595  1      $BITPOSITION ( %NAME ('GHB$V_',BITNAME) ),
M 0596  1      %ASCID MEANING
0597  1      %;
0598  1
0599  1 BIND
0600  1      PROTOCOLBITS =
0601  1      UPLIT
0602  1      (
P 0603  1      GHBERRBITS
P 0604  1          (
P 0605  1          START, '    Invalid start message received',
P 0606  1          CSBZERO, '    Zero node circuit index received',
P 0607  1          CSBRANGE, '    Node circuit index out of range',
P 0608  1          CSBINVALID, '    Node circuit sequence invalid',
P 0609  1          CSBSTALE, '    Node circuit index no longer valid',
P 0610  1          INVALIDSEQ, '    Invalid sequence number received in start message',
P 0611  1          HALT, '    Circuit was forced to halt',
P 0612  1          INVALIDREMID, '    Invalid server slot index',
P 0613  1          INVALIDLOCID, '    Invalid node slot index',
P 0614  1          BADCREDITS, '    Invalid credit field or too many credits used',
P 0615  1          REPCREATE, '    Repeat create of slot by server',
P 0616  1          REPDISC, '    Repeat disconnect of slot by server'
0617  1          ),
0618  1      0, 0
0619  1      );
0620  1
0621  1 !
0622  1 !  Define general error bit generation macro
0623  1 !
0624  1
M 0625  1 MACRO ERRBITS [BITVALUE, MEANING] =
M 0626  1      BITVALUE,
M 0627  1      %ASCID MEANING
0628  1      %;
0629  1
0630  1 !
```

```
495          0631  1 ! Build a list of Receive error bit values
496          0632  1 !
497          0633  1 BIND
498          0634  1     RCVERRBITS =
499          0635  1     UPLIT
500          0636  1     (
501        P 0637  1     ERRBITS
502        P 0638  1         (
503        P 0639  1         0, '    CRC errors',
504        P 0640  1         1, '    Framing errors',
505        P 0641  1         2, '    Frame length error'
506          0642  1         ),
507          0643  1     0, 0
508          0644  1     );
509          0645  1
510          0646  1 !
511          0647  1 ! Build a list of Transmit error bit values
512          0648  1 !
513          0649  1 BIND
514          0650  1     XMTERRBITS =
515          0651  1     UPLIT
516          0652  1     (
517        P 0653  1     ERRBITS
518        P 0654  1         (
519        P 0655  1         0, '    More than 16 retry failures',
520        P 0656  1         1, '    Loss of carrier',
521        P 0657  1         2, '    Unknown',
522        P 0658  1         3, '    Unknown',
523        P 0659  1         4, '    Frame length error',
524        P 0660  1         5, '    Late collision',
525          0661  1         ),
526          0662  1     0, 0
527          0663  1     );
528          0664  1
```

```
1 !
1 ! OWN STORAGE:
1 !
1
1 !
1 ! These are the strings and descriptors used to parse commands and qualifiers.
1 !
1 OWN
1     INPUT_LINE   : VECTOR [INPUT_LEN, BYTE],
1     INPUT_DSC    : BLOCK [DSC$K_Z_BLN, BYTE],
1     CMDBUF_DSC   : BLOCK [DSC$K_Z_BLN, BYTE],
1     SUBCMD       : VECTOR [CMDLEN, BYTE],
1     SUBCMDDSC    : BLOCK [DSC$K_Z_BLN, BYTE],
1     NAME         : VECTOR [GHB$K_NAMELEN, BYTE],
1     NAMEDSC      : BLOCK [DSC$K_Z_BLN, BYTE],
1     SERV         : VECTOR [GHB$K_NAMELEN, BYTE],
1     SERVDSC      : BLOCK [DSC$K_Z_BLN, BYTE],
1     ID           : VECTOR [GHB$K_IDLEN, BYTE],
1     IDDSC        : BLOCK [DSC$K_Z_BLN, BYTE],
1     NODE_ID      : VECTOR [GHB$K_IDLEN, BYTE],
1     NODE_IDDSC   : BLOCK [DSC$K_Z_BLN, BYTE],
1     NUMLST       : VECTOR [NUMLEN, BYTE],
1     NUMLSTDSC    : BLOCK [DSC$K_Z_BLN, BYTE],
1
1     GROUPS       : BLOCK [32, BYTE],
1     RATING       : LONG,
1     SERVRATING   : LONG,
1     VERSION      : LONG,
1     ECO          : LONG,
1     MCASTIMR     : LONG,
1
1     FLAGS        : BLOCK [1, LONG]                ! Flags word
1     ;
1
1 LITERAL OUTBUFSIZE = 1024,
1         GRPBUFSIZE = 1024;
1 !
1 ! These data structures are those used in starting, stopping and showing
1 ! the counters, characteristics and history buffer of the driver.
1 !
1
1 OWN
1     LCP$L_AREA : REF BLOCK [,BYTE],
1     AREA : BLOCK [MAX (GHB$S_COMM_AREA, GHB$K_HISTSIZE,
1                 (GHB$K_NAMELEN+LCB_C_LENGTH+1)*(GHB$K_ACT_CSBS+GHB$K_INACT_CSBS)),
1                    BYTE],
1     AREA_RET_SIZE : LONG,
1     OUTDSC : BLOCK [DSC$K_Z_BLN, BYTE],
1     OUTBUF : VECTOR [OUTBUFSIZE, BYTE],
1     GRPDSC : BLOCK [DSC$K_Z_BLN, BYTE],
1     GRPBUF : VECTOR [GRPBUFSIZE, BYTE]
1     ;
1
1
1 !
1 ! EXTERNAL REFERENCES:
1 !
```

```
1
1 EXTERNAL ROUTINE
1     CLI$GET_VALUE : ADDRESSING_MODE (GENERAL),
1     CLI$PRESENT  : ADDRESSING_MODE (GENERAL),
1     OTS$CVT_TI_L : ADDRESSING_MODE (GENERAL),
1     OTS$CVT_L_TI : ADDRESSING_MODE (GENERAL),
1     STR$TRIM : ADDRESSING_MODE (GENERAL),
1     STR$UPCASE : ADDRESSING_MODE (GENERAL),
1     LIB$PUT_OUTPUT : ADDRESSING_MODE (GENERAL),
1     LIB$FFS : ADDRESSING_MODE (GENERAL),
1     LIB$SKPC : ADDRESSING_MODE (GENERAL),
1     LIB$GET_INPUT : ADDRESSING_MODE (GENERAL),
1     LIB$GET_FOREIGN : ADDRESSING_MODE (GENERAL),
1     CLI$DCL_PARSE : ADDRESSING_MODE (GENERAL),
1     CLI$DISPATCH : ADDRESSING_MODE (GENERAL),
1     IOC$VERIFYCHAN : VERIFYCHAN ADDRESSING_MODE (GENERAL),
1     EXE$ALONONPAGED : ALONONPAGED ADDRESSING_MODE (GENERAL),
1     EXE$DEANONPAGED : DEANONPAGED ADDRESSING_MODE (GENERAL)
1     ;
1
1  EXTERNAL
1     LCP_COMMANDS,                          ! Command table
1     IOC$GL_DPTLIST : LONG                  ! HEAD OF LIST OF DPTS
1                 ADDRESSING_MODE (GENERAL)
1     ;
1
1 EXTERNAL LITERAL
1     CLI$_ABSENT,
1     LCP$_IVCMD,
1     LCP$_IVQUAL,
1     LCP$_NOTLOADED,
1     LCP$_NOTINITED,
1     LCP$_SET,
1     LCP$_NOTSET,
1     LCP$_STARTED,
1     LCP$_NOTSTARTED,
1     LCP$_STOPPED,
1     LCP$_NOTSTOPPED,
1     LCP$_NOHISTORY,
1     LCP$_NOSTARTHIST,
1     LCP$_NOTFROMLAT,
1     LCP$_NOSERVERS,
1     LCP$_INTERNAL
1     ;
1
```

```
%SBTTL  'LCP_MAIN  Main entry point'
ROUTINE LCP_MAIN =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Main entry point for LAT control program.  Obtain the command
!   and do the main dispatch.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       Foreign command read
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       exit status or signalled errors
!
! SIDE EFFECTS:
!
!       ltdriver controlled or information obtained.
!
!--

    BEGIN

    LOCAL
        STATUS
        ;

    !
    ! Go to see if the driver is loaded.  We will also fail early
    ! If we don't have CMKRNL priv.
    !
    IF NOT (STATUS = $CMKRNL (ROUTIN = LCPK_FINDDRIVER) )
    THEN
        RETURN .STATUS
    ;
    !
    !  Check for any foreign command
    !
    $SETDSC (CMDBUF_DSC, INPUT_LEN, INPUT_LINE);
    INPUT_DSC[DSC$A_POINTER] = INPUT_LINE;              ! Set address of buffer
    IF LIB$GET_FOREIGN(CMDBUF_DSC,0,INPUT_DSC)
        AND .INPUT_DSC [DSC$W_LENGTH] NEQ 0
    THEN
        BEGIN
        FLAGS [FLAG_L_ALL] = 0;                         ! Zero all flags
        STATUS = CLI$DCL_PARSE(INPUT_DSC, LCP_COMMANDS, LIB$GET_INPUT,
```

```
;  690          0824  3                          LIB$GET_INPUT, LCP_PROMPT);
;  691          0825  3          IF .STATUS
;  692          0826  3          THEN
;  693          0827  3              CLI$DISPATCH ();                          ! Dispatch on the VERB
;  694          0828  3
;  695          0829  3          RETURN SS$_NORMAL
;  696          0830  3
;  697          0831  3          END
;  698          0832  2      ;
;  699          0833  2
;  700          0834  3      UNTIL (STATUS = CLI$DCL_PARSE(0, LCP_COMMANDS, LIB$GET_INPUT,
;  701          0835  2                          LIB$GET_INPUT, LCP_PROMPT)) EQL RMS$_EOF
;  702          0836  2      DO
;  703          0837  2          IF .STATUS
;  704          0838  2          THEN
;  705          0839  2              CLI$DISPATCH ()
;  706          0840  2      ;
;  707          0841  2
;  708          0842  2      RETURN SS$_NORMAL
;  709          0843  2
;  710          0844  1      END;


                                                                .TITLE  LATCP LAT Control Program
                                                                .IDENT  \V04-000\

                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                          00  00  00  20  3E  50  43  4C  00000 P.AAB:  .ASCII  \LCP> \<0><0><0>
                                                  010E0005  00008 P.AAA:  .LONG   17694725
                                                  00000000' 0000C         .ADDRESS P.AAB
                      00  00  44  4E  41  4D  4D  4F  43  42  55  53  00010 P.AAD:  .ASCII  \SUBCOMMAND\<0><0>
                                                  010E000A  0001C P.AAC:  .LONG   17694730
                                                  00000000' 00020         .ADDRESS P.AAD
                      00  00  00  45  4D  41  4E  5F  45  44  4F  4E  00024 P.AAF:  .ASCII  \NODE_NAME\<0><0><0>
                                                  010E0009  00030 P.AAE:  .LONG   17694729
                                                  00000000' 00034         .ADDRESS P.AAF
46  49  54  4E  45  44  49  5F  45  43  49  56  52  45  53  00038 P.AAH:  .ASCII  \SERVICE_IDENTIFICATION\<0><0>
                        00  00  4E  4F  49  54  41  43  49  00047
                                                  010E0016  00050 P.AAG:  .LONG   17694742
                                                  00000000' 00054         .ADDRESS P.AAH
                          00  00  53  50  55  4F  52  47  00058 P.AAJ:  .ASCII  \GROUPS\<0><0>
                                                  010E0006  00060 P.AAI:  .LONG   17694726
                                                  00000000' 00064         .ADDRESS P.AAJ
                          00  00  47  4E  49  54  41  52  00068 P.AAL:  .ASCII  \RATING\<0><0>
                                                  010E0006  00070 P.AAK:  .LONG   17694726
                                                  00000000' 00074         .ADDRESS P.AAL
00  45  43  49  56  52  45  53  5F  47  4E  49  54  41  52  00078 P.AAN:  .ASCII  \RATING_SERVICE\<0><0>
                                                      00    00087
                                                  010E000E  00088 P.AAM:  .LONG   17694734
                                                  00000000' 0008C         .ADDRESS P.AAN
                      45  4D  41  4E  5F  45  43  49  56  52  45  53  00090 P.AAP:  .ASCII  \SERVICE_NAME\
                                                  010E000C  0009C P.AAO:  .LONG   17694732
                                                  00000000' 000A0         .ADDRESS P.AAP
                          00  00  00  54  52  41  54  53  000A4 P.AAR:  .ASCII  \START\<0><0><0>
                                                  010E0005  000AC P.AAQ:  .LONG   17694725
                                                  00000000' 000B0         .ADDRESS P.AAR
```

```
                                                50  4F  54  53  000B4 P.AAT:  .ASCII  \STOP\
                                                      010E0004  000B8 P.AAS:  .LONG   17694724
                                                      00000000' 000BC        .ADDRESS P.AAT
                                                57  4F  48  53  000C0 P.AAV:  .ASCII  \SHOW\
                                                      010E0004  000C4 P.AAU:  .LONG   17694724
                                                      00000000' 000C8        .ADDRESS P.AAV
                                                20  54  45  53  000CC P.AAX:  .ASCII  \SET \
                                                      010E0004  000D0 P.AAW:  .LONG   17694724
                                                      00000000' 000D4        .ADDRESS P.AAX
                                                4F  52  45  5A  000D8 P.AAZ:  .ASCII  \ZERO\
                                                      010E0004  000DC P.AAY:  .LONG   17694724
                                                      00000000' 000E0        .ADDRESS P.AAZ
                                53  52  45  54  4E  55  4F  43  000E4 P.ABB:  .ASCII  \COUNTERS\
                                                      010E0008  000EC P.ABA:  .LONG   17694728
                                                      00000000' 000F0        .ADDRESS P.ABB
                                00  59  52  4F  54  53  49  48  000F4 P.ABD:  .ASCII  \HISTORY\<0>
                                                      010E0007  000FC P.ABC:  .LONG   17694727
                                                      00000000' 00100        .ADDRESS P.ABD
53  43  49  54  53  49  52  45  54  43  41  52  41  48  43  00104 P.ABF:  .ASCII  \CHARACTERISTICS\<0>
                                                          00  00113
                                                      010E000F  00114 P.ABE:  .LONG   17694735
                                                      00000000' 00118        .ADDRESS P.ABF
52  45  4D  49  54  5F  54  53  41  43  49  54  4C  55  4D  0011C P.ABH:  .ASCII  \MULTICAST_TIMER\<0>
                                                          00  0012B
                                                      010E000F  0012C P.ABG:  .LONG   17694735
                                                      00000000' 00130        .ADDRESS P.ABH
                                00  53  52  45  56  52  45  53  00134 P.ABJ:  .ASCII  \SERVERS\<0>
                                                      010E0007  0013C P.ABI:  .LONG   17694727
                                                      00000000' 00140        .ADDRESS P.ABJ
                                00  00  45  43  49  56  45  44  00144 P.ABL:  .ASCII  \DEVICE\<0><0>
                                                      010E0006  0014C P.ABK:  .LONG   17694726
                                                      00000000' 00150        .ADDRESS P.ABL
                                                45  44  4F  4E  00154 P.ABN:  .ASCII  \NODE\
                                                      010E0004  00158 P.ABM:  .LONG   17694724
                                                      00000000' 0015C        .ADDRESS P.ABN
                                                00  47  4F  4C  00160 P.ABP:  .ASCII  \LOG\<0>
                                                      010E0003  00164 P.ABO:  .LONG   17694723
                                                      00000000' 00168        .ADDRESS P.ABP
41  43  49  46  49  54  4E  45  44  49  5F  45  44  4F  4E  0016C P.ABR:  .ASCII  \NODE_IDENTIFICATION\<0>
                                                  00  4E  4F  49  54  0017B
                                                      010E0013  00180 P.ABQ:  .LONG   17694739
                                                      00000000' 00184        .ADDRESS P.ABR
00  65  6C  62  61  6C  69  61  76  61  20  65  6E  6F  6E  00188 P.ABT:  .ASCII  \none available\<0><0>
                                                          00  00197
                                                      010E000E  00198 P.ABS:  .LONG   17694734
                                                      00000000' 0019C        .ADDRESS P.ABT
                                00  00  65  76  69  74  63  61  001A0 P.ABV:  .ASCII  \active\<0><0>
                                                      010E0006  001A8 P.ABU:  .LONG   17694726
                                                      00000000' 001AC        .ADDRESS P.ABV
                                65  76  69  74  63  61  6E  69  001B0 P.ABX:  .ASCII  \inactive\
                                                      010E0008  001B8 P.ABW:  .LONG   17694728
                                                      00000000' 001BC        .ADDRESS P.ABX
6E  75  6F  43  20  65  64  6F  4E  20  50  43  4C  2F  21  001C0 P.ABZ:  .ASCII  \!/LCP Node Counters!/!/!10UL   Receive f\
20  20  4C  55  30  31  21  2F  21  2F  21  73  72  65  74  001CF
                    66  20  65  76  69  65  63  65  52  20  001DE
20  20  20  4C  55  30  31  21  2F  21  73  65  6D  61  72  001E8        .ASCII  \rames!/!10UL   Receive errors!/!10UL   R\
21  73  72  6F  72  72  65  20  65  76  69  65  63  65  52  001F7
```

```
                    52 20 20 20 4C 55 30 31 21 2F  00206
74 61 63 69 6C      70 75 64 20 65 76 69 65 63 65  00210          .ASCII  \eceive duplicates!/!10UL   Transmit fram\
61 72 54 20 20      20 4C 55 30 31 21 2F 21 73 65  0021F
                    6D 61 72 66 20 74 69 6D 73 6E  0022E
61 72 54 20 20      20 4C 55 30 31 21 2F 21 73 65  00238          .ASCII  \es!/!10UL   Transmit errors!/!10XL   Las\
21 2F 21 73 72      6F 72 72 65 20 74 69 6D 73 6E  00247
                    73 61 4C 20 20 20 4C 58 30 31  00256
6C 69 61 66 20      74 69 6D 73 6E 61 72 74 20 74  00260          .ASCII  \t transmit failure code!/!10UL   Retrans\
4C 55 30 31 21      2F 21 65 64 6F 63 20 65 72 75  0026F
                    73 6E 61 72 74 65 52 20 20 20  0027E
4C 55 30 31 21      2F 21 73 6E 6F 69 73 73 69 6D  00288          .ASCII  \missiuns!/!10UL   Circuit timeouts!/!10U\
65 6D 69 74 20      74 69 75 63 72 69 43 20 20 20  00297
                    55 30 31 21 2F 21 73 74 75 6F  002A6
72 65 20 6C 6F      63 6F 74 6F 72 50 20 20 20 4C  002B0          .ASCII  \L   Protocol errors!/!10XL   Protocol bi\
50 20 20 20 4C      58 30 31 21 2F 21 73 72 6F 72  002BF
                    69 62 20 6C 6F 63 6F 74 6F 72  002CE
20 20 4C 55 30      31 21 2F 21 6B 73 61 6D 20 74  002D8          .ASCII  \t mask!/!10UL   Resource errors!/!10UL  \
72 6F 72 72 65      20 65 63 72 75 6F 73 65 52 20  002E7
                    20 20 4C 55 30 31 21 2F 21 73  002F6
75 62 20 74 69      6D 73 6E 61 72 74 20 6F 4E 20  00300          .ASCII  \ No transmit buffer!/!10UL   Unit timeou\
55 20 20 20 4C      55 30 31 21 2F 21 72 65 66 66  0030F
                    75 6F 65 6D 69 74 20 74 69 6E  0031E
                                       2F 21 73 74  00328          .ASCII  \ts!/\
                                          010E016C  0032C P.ABY:   .LONG   17695084
                                          00000000' 00330          .ADDRESS P.ABZ
72 72 65 20 6C      6F 63 6F 74 6F 72 50 20 0A 0D  00334 P.ACB:   .ASCII  <13><10>\ Protocol errors include:\<13>
      0D 3A 65 64   75 6C 63 6E 69 20 73 72 6F     00343
                                       00 00 00 0A  00350          .ASCII  <10><0><0><0>
                                          010E001D  00354 P.ACA:   .LONG   17694749
                                          00000000' 00358          .ADDRESS P.ACB
6F 43 20 72 65      76 72 65 53 20 50 43 4C 2F 21  0035C P.ACD:   .ASCII  \!/LCP Server Counters for !AS!/!/!10UL  \
21 53 41 21 20      72 6F 66 20 73 72 65 74 6E 75  0036B
                    20 20 4C 55 30 31 21 2F 21 2F  0037A
73 65 6D 61 72      66 20 65 76 69 65 63 65 52 20  00384          .ASCII  \ Receive frames!/!10UL   Transmit frames\
73 6E 61 72 54      20 20 20 4C 55 30 31 21 2F 21  00393
                    73 65 6D 61 72 66 20 74 69 6D  003A2
61 72 74 65 52      20 20 20 4C 55 30 31 21 2F 21  003AC          .ASCII  \!/!10UL   Retransmissions!/!10UL   Out o\
30 31 21 2F 21      73 6E 6F 69 73 73 69 6D 73 6E  003BB
                    6F 20 74 75 4F 20 20 20 4C 55  003CA
6D 61 72 66 20      65 63 6E 65 75 71 65 73 20 66  003D4          .ASCII  \f sequence frames!/!10UL   Invalid messa\
76 6E 49 20 20      20 4C 55 30 31 21 2F 21 73 65  003E3
                    61 73 73 65 6D 20 64 69 6C 61  003F2
6E 49 20 20 20      4C 55 30 31 21 2F 21 73 65 67  003FC          .ASCII  \ges!/!10UL   Invalid slots!/!/\<0><0>
2F 21 2F 21 73      74 6F 6C 73 20 64 69 6C 61 76  0040B
                                       00 00        0041A
                                          010E00BE  0041C P.ACC:   .LONG   17694910
                                          00000000' 00420          .ADDRESS P.ACD
20 74 65 6E 72      65 68 74 45 20 50 43 4C 2F 21  00424 P.ACF:   .ASCII  \!/LCP Ethernet Device Counters!/!/!10UL \
73 72 65 74 6E      75 6F 43 20 65 63 69 76 65 44  00433
                    20 4C 55 30 31 21 2F 21 2F 21  00442
65 63 6E 69 73      20 73 64 6E 6F 63 65 53 20 20  0044C          .ASCII  \  Seconds since last zeroed!/!10UL   Rec\
21 2F 21 64 65      6F 72 65 7A 20 74 73 61 6C 20  0045B
                    63 65 52 20 20 20 4C 55 30 31  0046A
31 21 2F 21 73      65 6D 61 72 66 20 65 76 69 65  00474          .ASCII  \eive frames!/!10UL   Receive errors!/!10\
65 20 65 76 69      65 63 65 52 20 20 20 4C 55 30  00483
                    30 31 21 2F 21 73 72 6F 72 72  00492
20 74 73 61 63      69 74 6C 75 4D 20 20 20 4C 55  0049C          .ASCII  \UL   Multicast frames received!/!10UL   \
```

```
64 65 76 69 65 63 65 72 20 73 65 6D 61 72 66  004AB
               20 20 20 4C 55 30 31 21 2F 21  004BA
2F 21 73 65 74 79 62 20 65 76 69 65 63 65 52  004C4        .ASCII  \Receive bytes!/!10UL   Multicast bytes r\
61 63 69 74 6C 75 4D 20 20 20 4C 55 30 31 21  004D3
               72 20 73 65 74 79 62 20 74 73  004E2
20 4C 55 30 31 21 2F 21 64 65 76 69 65 63 65  004EC        .ASCII  \eceived!/!10UL   Transmit frames!/!10UL \
6D 61 72 66 20 74 69 6D 73 6E 61 72 54 20 20  004FB
               20 4C 55 30 31 21 2F 21 73 65  0050A
6F 72 72 65 20 74 69 6D 73 6E 61 72 54 20 20  00514        .ASCII  \  Transmit errors!/!10UL   Multicast fra\
6C 75 4D 20 20 20 4C 55 30 31 21 2F 21 73 72  00523
               61 72 66 20 74 73 61 63 69 74  00532
64 65 74 74 69 6D 73 6E 61 72 74 20 73 65 6D  0053C        .ASCII  \mes transmitted!/!10UL   Transmit bytes!\
73 6E 61 72 54 20 20 20 4C 55 30 31 21 2F 21  0054B
               21 73 65 74 79 62 20 74 69 6D  0055A
63 69 74 6C 75 4D 20 20 20 4C 55 30 31 21 2F  00564        .ASCII  \/!10UL   Multicast bytes transmitted!/!1\
73 6E 61 72 74 20 73 65 74 79 62 20 74 73 61  00573
               31 21 2F 21 64 65 74 74 69 6D  00582
65 73 20 73 65 6D 61 72 46 20 20 20 4C 55 30  0058C        .ASCII  \0UL   Frames sent, single collision!/!10\
6C 6C 6F 63 20 65 6C 67 6E 69 73 20 2C 74 6E  0059B
               30 31 21 2F 21 6E 6F 69 73 69  005AA
6E 65 73 20 73 65 6D 61 72 46 20 20 20 4C 55  005B4        .ASCII  \UL   Frames sent, multiple collisions!/!\
6C 6F 63 20 65 6C 70 69 74 6C 75 6D 20 2C 74  005C3
               21 2F 21 73 6E 6F 69 73 69 6C  005D2
73 20 73 65 6D 61 72 46 20 20 20 4C 55 30 31  005DC        .ASCII  \10UL   Frames sent, initially deferred!/\
20 79 6C 6C 61 69 74 69 6E 69 20 2C 74 6E 65  005EB
               2F 21 64 65 72 72 65 66 65 64  005FA
69 6D 73 6E 61 72 54 20 20 20 4C 55 30 31 21  00604        .ASCII  \!10UL   Transmit collision detect check \
74 65 64 20 6E 6F 69 73 69 6C 6C 6F 63 20 74  00613
               20 6B 63 65 68 63 20 74 63 65  00622
20 4C 55 30 31 21 2F 21 65 72 75 6C 69 61 66  0062C        .ASCII  \failure!/!10UL   Data overrun!/!10UL   L\
21 6E 75 72 72 65 76 6F 20 61 74 61 44 20 20  0063B
               4C 20 20 20 4C 55 30 31 21 2F  0064A
72 72 65 20 72 65 66 66 75 62 20 6C 61 63 6F  00654        .ASCII  \ocal buffer errors!/!10UL   System buffe\
79 53 20 20 20 4C 55 30 31 21 2F 21 73 72 6F  00663
               65 66 66 75 62 20 6D 65 74 73  00672
2F 21 65 6C 62 61 6C 69 61 76 61 6E 75 20 72  0067C        .ASCII  \r unavailable!/!10UL   User buffer unava\
75 62 20 72 65 73 55 20 20 20 4C 55 30 31 21  0068B
               61 76 61 6E 75 20 72 65 66 66  0069A
                  2F 21 65 6C 62 61 6C 69     006A4        .ASCII  \ilable!/\
                                 010E0288     006AC P.ACE: .LONG   17695368
                                 00000000'    006B0        .ADDRESS P.ACF
                                 00000000     006B4 P.ACG: .LONG   0
                                 000003F2     006B8        .LONG   1010
                                 00000426     006BC        .LONG   1062
                                 000003F4     006C0        .LONG   1012
                                 000003E8     006C4        .LONG   1000
                                 000003EA     006C8        .LONG   1002
                                 000003F3     006CC        .LONG   1011
                                 00000424     006D0        .LONG   1060
                                 00000A8D     006D4        .LONG   2701
                                 000003E9     006D8        .LONG   1001
                                 00000A8E     006DC        .LONG   2702
                                 000003F6     006E0        .LONG   1014
                                 000003F7     006E4        .LONG   1015
                                 000003F5     006E8        .LONG   1013
                                 00000425     006EC        .LONG   1061
                                 00000428     006F0        .LONG   1064
```

```
                                                  00000411  006F4          .LONG   1041
                                                  00000429  006F8          .LONG   1065
                                                  0000042A  006FC          .LONG   1066
                                                  FFFFFFFF  00700          .LONG   -1
6F  72  72  65  20  65  76  69  65  63  65  52  20  0A  0D  00704 P.ACI:   .ASCII  <13><10>\ Receive errors include:\<13>
            0D  3A  65  64  75  6C  63  6E  69  20  73  72  00713
                                                        0A  0071F          .ASCII  <10>
                                                  010E001C  00720 P.ACH:   .LONG   17694748
                                                  00000000' 00724          .ADDRESS P.ACI
72  72  65  20  74  69  6D  73  6E  61  72  54  20  0A  0D  00728 P.ACK:   .ASCII  <13><10>\ Transmit errors include:\<13>
        0D  3A  65  64  75  6C  63  6E  69  20  73  72  6F  00737
                                                00  00  00  0A  00744          .ASCII  <10><0><0><0>
                                                  010E001D  00748 P.ACJ:   .LONG   17694749
                                                  00000000' 0074C          .ADDRESS P.ACK
20  20  20  20  65  6D  69  54  20  20  20  20  20  2F  21  00750 P.ACM:   .ASCII  \!/     Time       Type Slots Idx Seq   I\
49  20  73  74  6F  6C  53  20  65  70  79  54  20  20  20  0075F
                    49  20  20  20  71  65  53  20  78  64  0076E
63  41  20  71  65  53  20  20  20  71  65  53  20  78  64  00778          .ASCII  \dx Seq   Seq Ack    DATA...!/\<0><0>
00  2F  21  2E  2E  2E  41  54  41  44  20  20  20  20  6B  00787
                                                        00  00796
                                                        00  00797          .ASCII  <0>
                                                  010E0045  00798 P.ACL:   .LONG   17694789
                                                  00000000' 0079C          .ADDRESS P.ACM
42  58  21  20  20  3E  21  54  25  21  3C  34  31  21  20  007A0 P.ACO:   .ASCII  \ !14<!%T!>  !XB   !XB    !XB  !XB    !XB\
20  20  42  58  21  20  20  20  20  42  58  21  20  20  20  007AF
                    42  58  21  20  20  20  20  42  58  21  007BE
58  21  20  42  58  21  20  20  20  20  42  58  21  20  20  007C8          .ASCII  \  !XB    !XB !XB     !XB !XB  !XB !XB  !\
20  20  42  58  21  20  42  58  21  20  20  20  20  20  42  007D7
                    21  20  20  42  58  21  20  42  58  21  007E6
42  58  21  20  42  58  21  20  20  42  58  21  20  42  58  007F0          .ASCII  \XB !XB  !XB !XB  !XB !XB  !XB !XB  !XB !\
20  42  58  21  20  20  42  58  21  20  42  58  21  20  20  007FF
                    21  20  42  58  21  20  20  42  58  21  0080E
00  2F  21  20  42  58  21  20  42  58  21  20  20  42  58  00818          .ASCII  \XB  !XB !XB !/\<0><0>
                                                        00  00827
                                                  010E0086  00828 P.ACN:   .LONG   17694854
                                                  00000000' 0082C          .ADDRESS P.ACO
            00  00  45  43  49  56  45  44  24  54  41  4C  00830 P.ACR:   .ASCII  \LAT$DEVICE\<0><0>
                                                  010E000A  0083C P.ACQ:   .LONG   17694730
                                                  00000000' 00840          .ADDRESS P.ACR
                        00  00  00  3A  30  41  45  58  00844 P.ACT:   .ASCII  \XEA0:\<0><0><0>
                                                  010E0005  0084C P.ACS:   .LONG   17694725
                                                  00000000' 00850          .ADDRESS P.ACT
                        00  00  00  3A  30  41  51  58  00854 P.ACV:   .ASCII  \XQA0:\<0><0><0>
                                                  010E0005  0085C P.ACU:   .LONG   17694725
                                                  00000000' 00860          .ADDRESS P.ACV
                          00000000' 00000000' 00000000' 00864 P.ACP:   .ADDRESS P.ACQ, P.ACS, P.ACU
                                                  00000000  00870          .LONG   0
                                                      0AF1  00874 P.ACW:   .WORD   2801
                                                  000005DC  00876          .LONG   1500
                                                      0451  0087A          .WORD   1105
                                                  00000001  0087C          .LONG   1
                                                      0B1A  00880          .WORD   2842
                                                  00000001  00882          .LONG   1
                                                      0B0E  00886          .WORD   2830
                                                  00000460  00888          .LONG   1120
                                                      0B04  0088C          .WORD   2820
                                                      0002  0088E          .WORD   2
```

```
                                                  0004  00890          .WORD   4
                                                  0B18  00892          .WORD   2840
                                            00000001  00894          .LONG   1
                                                  0B1B  00898          .WORD   2843
                                            00000001  0089A          .LONG   1
                                                  0B1C  0089E          .WORD   2844
                                            00000000  008A0          .LONG   0
                                                  0456  008A4          .WORD   1110
                                            00000000  008A6          .LONG   0
                                                    00  008AA P.ACX:   .BYTE   0
                                                        008AB          .BLKB   1
                                            00000036  008AC P.ACY:   .LONG   54
                                            00000000' 008B0          .ADDRESS SETPARM
61  74  73  20  64  69  6C  61  76  6E  49  20  20  20  20  008B4 P.ADB:   .ASCII  \    Invalid start message received\<0>
65  63  65  72  20  65  67  61  73  73  65  6D  20  74  72  008C3
                                        00  64  65  76  69  008D2
                                                    00  008D7          .ASCII  <0>
                                            010E0022  008D8 P.ADA:   .LONG   17694754
                                            00000000' 008DC          .ADDRESS P.ADB
63  20  65  64  6F  6E  20  6F  72  65  5A  20  20  20  20  008E0 P.ADD:   .ASCII  \    Zero node circuit index received\
65  72  20  78  65  64  6E  69  20  74  69  75  63  72  69  008EF
                                    64  65  76  69  65  63  008FE
                                            010E0024  00904 P.ADC:   .LONG   17694756
                                            00000000' 00908          .ADDRESS P.ADD
69  75  63  72  69  63  20  65  64  6F  4E  20  20  20  20  0090C P.ADF:   .ASCII  \    Node circuit index out of range\<0>
20  66  6F  20  74  75  6F  20  78  65  64  6E  69  20  74  0091B
                                    00  65  67  6E  61  72  0092A
                                            010E0023  00930 P.ADE:   .LONG   17694755
                                            00000000' 00934          .ADDRESS P.ADF
69  75  63  72  69  63  20  65  64  6F  4E  20  20  20  20  00938 P.ADH:   .ASCII  \    Node circuit sequence invalid\<0><0>
61  76  6E  69  20  65  63  6E  65  75  71  65  73  20  74  00947
                                        00  00  64  69  6C  00956
                                                    00  0095B          .ASCII  <0>
                                            010E0021  0095C P.ADG:   .LONG   17694753
                                            00000000' 00960          .ADDRESS P.ADH
69  75  63  72  69  63  20  65  64  6F  4E  20  20  20  20  00964 P.ADJ:   .ASCII  \    Node circuit index no longer valid-
67  6E  6F  6C  20  6F  6E  20  78  65  64  6E  69  20  74  00973                \<0>
                            00  64  69  6C  61  76  20  72  65  00982
                                                    00  0098B          .ASCII  <0>
                                            010E0026  0098C P.ADI:   .LONG   17694758
                                            00000000' 00990          .ADDRESS P.ADJ
71  65  73  20  64  69  6C  61  76  6E  49  20  20  20  20  00994 P.ADL:   .ASCII  \    Invalid sequence number received in \
65  72  20  72  65  62  6D  75  6E  20  65  63  6E  65  75  009A3
                        20  6E  69  20  64  65  76  69  65  63  009B2
00  00  65  67  61  73  73  65  6D  20  74  72  61  74  73  009BC          .ASCII  \start message\<0><0><0>
                                                    00  009CB
                                            010E0035  009CC P.ADK:   .LONG   17694773
                                            00000000' 009D0          .ADDRESS P.ADL
73  61  77  20  74  69  75  63  72  69  43  20  20  20  20  009D4 P.ADN:   .ASCII  \    Circuit was forced to halt\<0><0>
74  6C  61  68  20  6F  74  20  64  65  63  72  6F  66  20  009E3
                                                    00  00  009F2
                                            010E001E  009F4 P.ADM:   .LONG   17694750
                                            00000000' 009F8          .ADDRESS P.ADN
72  65  73  20  64  69  6C  61  76  6E  49  20  20  20  20  009FC P.ADP:   .ASCII  \    Invalid server slot index\<0><0>
00  78  65  64  6E  69  20  74  6F  6C  73  20  72  65  76  00A0B
                                                    00  00A1A
                                                    00  00A1B          .ASCII  <0>
```

```
                                                    010E001D  00A1C P.ADO:  .LONG    17694749
                                                    00000000' 00A20         .ADDRESS P.ADP
64 6F 6E 20 64 69 6C 61 76 6E 49 20 20 20 20 20     00A24 P.ADR:  .ASCII  \     Inva id node slot index\<0>
        00 78 65 64 6E 69 20 74 6F 6C 73 20 65      00A33
                                                    010E001B  00A40 P.ADQ:  .LONG    17694747
                                                    00000000' 00A44         .ADDRESS P.ADR
65 72 63 20 64 69 6C 61 76 6E 49 20 20 20 20 20     00A48 P.ADT:  .ASCII  \     Invalid credit field or too many cre\
6F 74 20 72 6F 20 64 6C 65 69 66 20 74 69 64       00A57
                  65 72 63 20 79 6E 61 6D 20 6F     00A66
              00 00 00 64 65 73 75 20 73 74 69 64   00A70         .ASCII  \dits used\<0><0><0>
                                                    010E0031  00A7C P.ADS:  .LONG    17694769
                                                    00000000' 00A80         .ADDRESS P.ADT
61 65 72 63 20 74 61 65 70 65 52 20 20 20 20 20     00A84 P.ADV:  .ASCII  \     Repeat create of slot by server\<0>
73 20 79 62 20 74 6F 6C 73 20 66 6F 20 65 74       00A93
                                 00 72 65 76 72 65  00AA2
                                                    010E0023  00AA8 P.ADU:  .LONG    17694755
                                                    00000000' 00AAC         .ADDRESS P.ADV
63 73 69 64 20 74 61 65 70 65 52 20 20 20 20 20     00AB0 P.ADX:  .ASCII  \     Repeat disconnect of slot by server-
20 74 6F 6C 73 20 66 6F 20 74 63 65 6E 6E 6F        00ABF                 \<0>
                  00 72 65 76 72 65 73 20 79 62     00ACE
                                                    010E0027  00AD8 P.ADW:  .LONG    17694759
                                                    00000000' 00ADC         .ADDRESS P.ADX
                                                    00000000  00AE0 P.ACZ:  .LONG    0
                                                    00000000' 00AE4         .ADDRESS P.ADA
                                                    00000001  00AE8         .LONG    1
                                                    00000000' 00AEC         .ADDRESS P.ADC
                                                    00000002  00AF0         .LONG    2
                                                    00000000' 00AF4         .ADDRESS P.ADE
                                                    00000003  00AF8         .LONG    3
                                                    00000000' 00AFC         .ADDRESS P.ADG
                                                    00000004  00B00         .LONG    4
                                                    00000000' 00B04         .ADDRESS P.ADI
                                                    0000000A  00B08         .LONG    10
                                                    00000000' 00B0C         .ADDRESS P.ADK
                                                    00000005  00B10         .LONG    5
                                                    00000000' 00B14         .ADDRESS P.ADM
                                                    00000006  00B18         .LONG    6
                                                    00000000' 00B1C         .ADDRESS P.ADO
                                                    00000007  00B20         .LONG    7
                                                    00000000' 00B24         .ADDRESS P.ADQ
                                                    00000008  00B28         .LONG    8
                                                    00000000' 00B2C         .ADDRESS P.ADS
                                                    00000009  00B30         .LONG    9
                                                    00000000' 00B34         .ADDRESS P.ADU
                                                    0000000B  00B38         .LONG    11
                                                    00000000' 00B3C         .ADDRESS P.ADW
                                        00000000    00000000  00B40         .LONG    0, 0
00 73 72 6F 72 72 65 20 43 52 43 20 20 20 20        00B48 P.AEA:  .ASCII  \     CRC errors\<0><0>
                                                 00 00B57
                                                    010E000E  00B58 P.ADZ:  .LONG    17694734
                                                    00000000' 00B5C         .ADDRESS P.AEA
72 72 65 20 67 6E 69 6D 61 72 46 20 20 20 20        00B60 P.AEC:  .ASCII  \     Framing errors\<0><0>
                                 00 00 73 72 6F     00B6F
                                                    010E0012  00B74 P.AEB:  .LONG    17694738
                                                    00 00000' 00B78         .ADDRESS P.AEC
74 67 6E 65 6C 20 65 6D 61 72 46 20 20 20 20        00B7C P.AEE:  .ASCII  \     Frame length error\<0><0>
                  00 00 72 6F 72 72 65 20 68        00B8B
```

```
                                                    010E0016  00B94 P.AED:  .LONG    17694742
                                                    00000000' 00B98         .ADDRESS P.AEE
                                                    00000000  00B9C P.ADY:  .LONG    0
                                                    00000000' 00BA0         .ADDRESS P.ADZ
                                                    00000001  00BA4         .LONG    1
                                                    00000000' 00BA8         .ADDRESS P.AEB
                                                    00000002  00BAC         .LONG    2
                                                    00000000' 00BB0         .ADDRESS P.AED
                                         00000000   00000000  00BB4         .LONG    0, 0
31  20  6E  61  68  74  20  65  72  6F  4D  20  20  20  20  00BBC P.AEH:  .ASCII  \    More than 16 retry failures\<0>
65  72  75  6C  69  61  66  20  79  72  74  65  72  20  36  00BCB
                                                    00  73  00BDA
                                                    010E001F  00BDC P.AEG:  .LONG    17694751
                                                    00000000' 00BE0         .ADDRESS P.AEH
72  61  63  20  66  6F  20  73  73  6F  4C  20  20  20  20  00BE4 P.AEJ:  .ASCII  \    Loss of carrier\<0>
                                         00  72  65  69  72  00BF3
                                                    010E0013  00BF8 P.AEI:  .LONG    17694739
                                                    00000000' 00BFC         .ADDRESS P.AEJ
            00  6E  77  6F  6E  6B  6E  55  20  20  20  20  00C00 P.AEL:  .ASCII  \    Unknown\<0>
                                                    010E000B  00C0C P.AEK:  .LONG    17694731
                                                    00000000' 00C10         .ADDRESS P.AEL
            00  6E  77  6F  6E  6B  6E  55  20  20  20  20  00C14 P.AEN:  .ASCII  \    Unknown\<0>
                                                    010E000B  00C20 P.AEM:  .LONG    17694731
                                                    00000000' 00C24         .ADDRESS P.AEN
74  67  6E  65  6C  20  65  6D  61  72  46  20  20  20  20  00C28 P.AEP:  .ASCII  \    Frame length error\<0><0>
                        00  00  72  6F  72  72  65  20  68  00C37
                                                    010E0016  00C40 P.AEO:  .LONG    17694742
                                                    00000000' 00C44         .ADDRESS P.AEP
73  69  6C  6C  6F  63  20  65  74  61  4C  20  20  20  20  00C48 P.AER:  .ASCII  \    Late collision\<0><0>
                                         00  00  6E  6F  69  00C57
                                                    010E0012  00C5C P.AEQ:  .LONG    17694738
                                                    00000000' 00C60         .ADDRESS P.AER
                                                    00000000  00C64 P.AEF:  .LONG    0
                                                    00000000' 00C68         .ADDRESS P.AEG
                                                    00000001  00C6C         .LONG    1
                                                    00000000' 00C70         .ADDRESS P.AEI
                                                    00000002  00C74         .LONG    2
                                                    00000000' 00C78         .ADDRESS P.AEK
                                                    00000003  00C7C         .LONG    3
                                                    00000000' 00C80         .ADDRESS P.AEM
                                                    00000004  00C84         .LONG    4
                                                    00000000' 00C88         .ADDRESS P.AEO
                                                    00000005  00C8C         .LONG    5
                                                    00000000' 00C90         .ADDRESS P.AEQ
                                         00000000   00000000  00C94         .LONG    0, 0

                                                                            .PSECT  $OWN$,NOEXE,2

                                                    00000 INPUT_LINE:
                                                                            .BLKB   1024
                                                    00400 INPUT_DSC:
                                                                            .BLKB   8
                                                    00408 CMDBUF_DSC:
                                                                            .BLKB   8
                                                    00410 SUBCMD: .BLKB     16
                                                    00420 SUBCMDDSC:
                                                                            .BLKB   8
```

```
00428 NAME:     .BLKB   16
00438 NAMEDSC:.BLKB   8
00440 SERV:     .BLKB   16
00450 SERVDSC:.BLKB   8
00458 ID:       .BLKB   64
00498 IDDSC:    .BLKB   8
004A0 NODE_ID:.BLKB   64
004E0 NODE_IDDSC:
                .BLKB   8
004E8 NUMLST: .BLKB   16
004F8 NUMLSTDSC:
                .BLKB   8
00500 GROUPS: .BLKB   32
00520 RATING: .BLKB   4
00524 SERVRATING:
                .BLKB   4
00528 VERSION:.BLKB   4
0052C ECO:      .BLKB   4
00530 MCASTIMR:
                .BLKB   4
00534 FLAGS:    .BLKB   4
00538 LCP$L_AREA:
                .BLKB   4
0053C AREA:     .BLKB   2560
00F3C AREA_RET_SIZE:
                .BLKB   4
00F40 OUTDSC: .BLKB   8
00F48 OUTBUF: .BLKB   1024
01348 GRPDSC: .BLKB   8
01350 GRPBUF: .BLKB   1024

      LCP_PROMPT=           P.AAA
      QUAL_SUBCOMMAND=      P.AAC
      QUAL_NAME=            P.AAE
      QUAL_ID=              P.AAG
      QUAL_GROUPS=          P.AAI
      QUAL_RATING=          P.AAK
      QUAL_SERV_RATING=     P.AAM
      QUAL_SERV_NAME=       P.AAO
      QUAL_START=           P.AAQ
      QUAL_STOP=            P.AAS
      QUAL_SHOW=            P.AAU
      QUAL_SET=             P.AAW
      QUAL_ZERO=            P.AAY
      QUAL_COUNTERS=        P.ABA
      QUAL_HISTORY=         P.ABC
      QUAL_CHAR=            P.ABE
      QUAL_MCAST=           P.ABG
      QUAL_SERVERS=         P.ABI
      QUAL_DEVICE=          P.ABK
      QUAL_NODE=            P.ABM
      QUAL_LOG=             P.ABO
      QUAL_NODEID=          P.ABQ
      QUAL_NONE=            P.ABS
      ACTIVE=               P.ABU
      INACTIVE=             P.ABW
      COUNTERSCTRSTR=       P.ABY
```

```
                                PROTOCOLMASK=       P.ACA
                                SERVCTRSTR=         P.ACC
                                DEVCTRSTR=          P.ACE
                                CTRNAMES=           P.ACG
                                RCVERRMASK=         P.ACH
                                XMTERRMASK=         P.ACJ
                                FAOSTR1=            P.ACL
                                FAOSTR2=            P.ACN
                                DEVNAMES=           P.ACP
                                SETPARM=            P.ACW
                                SETPARMEND=         P.ACX
                                SETPARMDSC=         P.ACY
                                PROTOCOLBITS=       P.ACZ
                                RCVERRBITS=         P.ADY
                                XMTERRBITS=         P.AEF
                                        .EXTRN  CLI$GET_VALUE, CLI$PRESENT
                                        .EXTRN  OTS$CVT_TI_L, OTS$CVT_L_TI
                                        .EXTRN  STR$TRIM, STR$UPCASE
                                        .EXTRN  LIB$PUT_OUTPUT, LIB$FFS
                                        .EXTRN  LIB$SKPC, LIB$GET_INPUT
                                        .EXTRN  LIB$GET_FOREIGN
                                        .EXTRN  CLI$DCL_PARSE, CLI$DISPATCH
                                        .EXTRN  IOC$VERIFYCHAN, EXE$ALONONPAGED
                                        .EXTRN  EXE$DEANONPAGED
                                        .EXTRN  LCP_COMMANDS, IOC$GL_DPTLIST
                                        .EXTRN  CLI$_ABSENT, LCP$_IVCMD
                                        .EXTRN  LCP$_IVQUAL, LCP$_NOTLOADED
                                        .EXTRN  LCP$_NOTINITED, LCP$_SET
                                        .EXTRN  LCP$_NOTSET, LCP$_STARTED
                                        .EXTRN  LCP$_NOTSTARTED
                                        .EXTRN  LCP$_STOPPED, LCP$_NOTSTOPPED
                                        .EXTRN  LCP$_NOHISTORY, LCP$_NOSTARTHIST
                                        .EXTRN  LCP$_NOTFROMLAT
                                        .EXTRN  LCP$_NOSERVERS, LCP$_INTERNAL
                                        .EXTRN  SYS$CMKRNL

                                        .PSECT  $CODE$,NOWRT,2

                            007C 00000 LCP_MAIN:
                                        .WORD   Save R2,R3,R4,R5,R6                     ; 0768
                 56 00000000G  00  9E 00002     MOVAB   CLI$DISPATCH, R6                ;
                 55 00000000G  00  9E 00009     MOVAB   CLI$DCL_PARSE, R5               ;
                 54 00000000G  00  9E 00010     MOVAB   LIB$GET_INPUT, R4               ;
                 53      0000'  CF  9E 00017    MOVAB   INPUT_DSC, R3                   ;
                                7E  D4 0001C    CLRL    -(SP)                           ; 0809
                         0000V  CF  9F 0001E    PUSHAB  LCPK_FINDDRIVER                 ;
        00000000G  00           02  FB 00022    CALLS   #2, SYS$CMKRNL                  ;
                 52             50  D0 00029    MOVL    R0, STATUS                      ;
                 04             52  E8 0002C    BLBS    STATUS, 1$                      ;
                 50             52  D0 0002F    MOVL    STATUS, R0                      ; 0811
                                    04 00032    RET                                     ;
              08 A3      0400  8F  3C 00033 1$: MOVZWL  #1024, CMDBUF_DSC               ; 0816
              0C A3      FC00  C3  9E 00039     MOVAB   INPUT_LINE, CMDBUF_DSC+4        ;
              04 A3      FC00  C3  9E 0003F     MOVAB   INPUT_LINE, INPUT_DSC+4         ; 0817
                                53  DD 00045    PUSHL   R3                              ; 0818
                                7E  D4 00047    CLRL    -(SP)                           ;
                           08   A3  9F 00049    PUSHAB  CMDBUF_DSC                      ;
```

```
                00000000G  00          03  FB 0004C          CALLS   #3, LIB$GET_FOREIGN                  ;
                           24          50  E9 00053          BLBC    R0, 2$                             ;
                                       63  B5 00056          TSTW    INPUT_DSC                          ; 0819
                                       20  13 00058          BEQL    2$                                 ;
                                 0134  C3  D4 0005A          CLRL    FLAGS                              ; 0822
                                 0000' CF  9F 0005E          PUSHAB  LCP_PROMPT                         ; 0823
                                       54  DD 00062          PUSHL   R4                                 ;
                                       54  DD 00064          PUSHL   R4                                 ;
                                 0000G CF  9F 00066          PUSHAB  LCP_COMMANDS                       ;
                                       53  DD 0006A          PUSHL   R3                                 ;
                           65          05  FB 0006C          CALLS   #5, CLI$DCL_PARSE                  ;
                           52          50  D0 0006F          MOVL    R0, STATUS                         ;
                           2A          52  E9 00072          BLBC    STATUS, 3$                         ; 0825
                           66          00  FB 00075          CALLS   #0, CLI$DISPATCH                   ; 0827
                                       25  11 00078          BRB     3$                                 ; 0829
                                 0000' CF  9F 0007A 2$:      PUSHAB  LCP_PROMPT                         ; 0834
                                       54  DD 0007E          PUSHL   R4                                 ;
                                       54  DD 00080          PUSHL   R4                                 ;
                                 0000G CF  9F 00082          PUSHAB  LCP_COMMANDS                       ;
                                       7E  D4 00086          CLRL    -(SP)                              ;
                           65          05  FB 00088          CALLS   #5, CLI$DCL_PARSE                  ;
                           52          50  D0 0008B          MOVL    R0, STATUS                         ;
                0001827A   8F          52  D1 0008E          CMPL    STATUS, #98938                     ; 0835
                                       08  13 00095          BEQL    3$                                 ;
                           E0          52  E9 00097          BLBC    STATUS, 2$                         ; 0837
                           66          00  FB 0009A          CALLS   #0, CLI$DISPATCH                   ; 0839
                                       DB  11 0009D          BRB     2$                                 ; 0837
                           50          01  D0 0009F 3$:      MOVL    #1, R0                             ; 0842
                                           04 000A2          RET                                        ; 0844

; Routine Size:  163 bytes,    Routine Base:  $CODE$ + 0000
```

```
%SBTTL  'LCPK_FINDDRIVER  Find the address of the communication area'
ROUTINE LCPK_FINDDRIVER =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Find the address of LTDRIVER and return the address as the value of
!   this routine, or return failure.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       ioc$gl_dptlist
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       address of the communication driver
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    !
    ! obtain the address of the ltdriver dpt and return the address of
    ! the counter block.  If no ltdriver, then return zero.
    !

    !
    ! size and address of the driver name string
    !
    BIND
        DRVRNAM = DESCRIPTOR ('LTDRIVER') : BLOCK [,BYTE]
        ;

    LOCAL
        DPT : REF BLOCK [,BYTE],
        DPTEND : LONG
        ;
    !
    ! scan the dpt list looking for ltdriver by name.
    !
    DPT = .IOC$GL_DPTLIST;
    DPTEND = IOC$GL_DPTLIST;
    WHILE .DPT NEQ .DPTEND
    DO
```

```
;  769          0902  3          BEGIN
;  770          0903  3          IF
;  771          0904  3              CH$EQL
;  772          0905  3                  (
;  773          0906  3                  .DRVRNAM [DSC$W_LENGTH], .DRVRNAM [DSC$A_POINTER],
;  774          0907  3                  CH$RCHAR (DPT [DPT$T_NAME]),
;  775          0908  3                  DPT [DPT$T_NAME] +1,
;  776          0909  3                  0
;  777          0910  3                  )
;  778          0911  3          THEN
;  779          0912  4              BEGIN
;  780          0913  4              !
;  781          0914  4              ! address of the counters is found in the longword
;  782          0915  4              ! preceeding the vector.
;  783          0916  4              !
;  784          0917  5              IF (LCP$L_AREA = .( (.DPT + .DPT [DPT$W_VECTOR]) - 4) )
;  785          0918  4                  LSS 0
;  786          0919  4              THEN
;  787          0920  4                  RETURN SUCCESS
;  788          0921  4              ELSE
;  789          0922  4                  RETURN LCP$_NOTINITED
;  790          0923  4              END
;  791          0924  3          ;
;  792          0925  3          DPT = .DPT [DPT$L_FLINK]
;  793          0926  3          END
;  794          0927  2      ;
;  795          0928  2      RETURN LCP$_NOTLOADED
;  796          0929  2
;  797          0930  1      END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                52  45  56  49  52  44  54  4C  00C9C P.AET:  .ASCII  \LTDRIVER\                                  ;
                                    00000008  00CA4 P.AES:  .LONG   8                                            ;
                                    00000000' 00CA8         .ADDRESS P.AET                                       ;

                                                    DRVRNAM=            P.AES


                                                        .PSECT  $CODE$,NOWRT,2

                                          007C 00000 LCPK_FINDDRIVER:
                                                        .WORD   Save R2,R3,R4,R5,R6                              ; 0846
                        56 00000000G  00  9E 00002      MOVAB   IOC$GL_DPTLIST, R6                               ;
                        54            66  D0 00009      MOVL    IOC$GL_DPTLIST1, DPT                             ; 0898
                        55            66  9E 0000C      MOVAB   IOC$GL_DPTLIST, DPTEND                           ; 0899
                        55            54  D1 0000F 1$:  CMPL    DPT, DPTEND                                      ; 0900
                                      31  13 00012      BEQL    4$                                               ;
                        50       20   A4  9A 00014      MOVZBL  32(DPT), R0                                      ; 0907
50          00    0000' DF     0000' CF  2D 00018      CMPC5   DRVRNAM, @DRVRNAM+4, #0, R0, 33(DPT)             ;
                                 21   A4     00021                                                               ;
                                      1B  12 00023      BNEQ    3$                                               ;
                        50       1E   A4  3C 00025      MOVZWL  30(DPT), R0                                      ; 0917
                                 FC A044  9F 00029      PUSHAB  -4(R0)[DPT]                                      ;
                    0000' CF          9E  D0 0002D      MOVL    @(SP)+, LCP$L_AREA                               ;
```

```
                                 04  18 00032          BGEQ    2$                                  ; 0918
                        50       01  D0 00034          MOVL    #1, R0                              ; 0922
                                     04 00037          RET                                         ;
                        50 00000000G 8F  D0 00038 2$:  MOVL    #LCP$_NOTINITED, R0                 ;
                                     04 0003F          RET                                         ;
                        54           64  D0 00040 3$:  MOVL    (DPT), DPT                          ; 0925
                                     CA  11 00043      BRB     1$                                  ;
                        50 00000000G 8F  D0 00045 4$:  MOVL    #LCP$_NOTLOADED, R0                 ; 0928
                                     04 0004C          RET                                         ; 0930

; Routine Size:  77 bytes,    Routine Base:  $CODE$ + 00A3
```

```
1 %SBTTL  'LCP_QUAL  Process multicast qualifiers'
1 ROUTINE LCP_QUAL =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !   Process qualifiers for the multicast message.  Convert the qualifiers
1 !   to descriptors to set in the driver.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       cli processed qualifiers
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       qualifier descriptors
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       errors signalled
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     LOCAL
2         BITVALUE,
2         PTR      : REF VECTOR [,BYTE],
2         DSC      : BLOCK [DSC$K_Z_BLN, BYTE],
2         STATUS
2         ;
2
2     !
2     ! Convert the name to a string and require that we get something
2     !
2     IF  NOT LCP_GETSTR (QUAL_NAME, NAMEDSC, GHB$K_NAMELEN, NAME,
2                 %ASCID 'SYS$NODE', SYI$_NODENAME)
2         OR
2         NOT LCP_FIXNAME (NAMEDSC)
2     THEN
3         BEGIN
3         SIGNAL (LCP$_IVQUAL, 2, QUAL_NAME, NAMEDSC);
3         RETURN FAILURE
3         END
2     ;
2
2     !
2     !  The node id is optional
```

```
;  856          0988  2      !
;  857          0989  2      LCP_GETSTR (QUAL_NODEID, NODE_IDDSC, GHB$K_IDLEN, NODE_ID,
;  858          0990  2          %ASCID 'SYS$ANNOUNCE', 0);
;  859          0991  2
;  860          0992  2      !
;  861          0993  2      ! The service name is optional
;  862          0994  2      !
;  863          0995  2      IF  LCP_GETSTR (QUAL_SERV_NAME, SERVDSC, GHB$K_NAMELEN, SERV,
;  864          0996  2          %ASCID 'SYS$SERVICE_NAME', 0)
;  865          0997  2      THEN
;  866          0998  3          BEGIN
;  867          0999  3          IF NOT LCP_FIXNAME (SERVDSC)
;  868          1000  3          THEN
;  869          1001  4              BEGIN
;  870          1002  4              SIGNAL (LCP$_IVQUAL, 2, QUAL_SERV_NAME, SERVDSC);
;  871          1003  4              RETURN FAILURE
;  872          1004  4              END
;  873          1005  3          END
;  874          1006  2      ;
;  875          1007  2
;  876          1008  2      !
;  877          1009  2      ! The service id string is optional too
;  878          1010  2      !
;  879          1011  2      LCP_GETSTR (QUAL_ID, IDDSC, GHB$K_IDLEN, ID, %ASCID 'SYS$ANNOUNCE', 0);
;  880          1012  2
;  881          1013  2      !
;  882          1014  2      ! Convert the groups as a list of items
;  883          1015  2      !
;  884          1016  2      $SETDSC (DSC, NUMLEN, NUMLST);
;  885          1017  2      STATUS = CLI$GET_VALUE (QUAL_GROUPS, DSC);
;  886          1018  2      !
;  887          1019  2      ! If there is no QUALIFIER for GROUPS,
;  888          1020  2      ! Then we will default to enable group code 0
;  889          1021  2      !
;  890          1022  2      IF NOT .STATUS
;  891          1023  2      THEN
;  892          1024  2          GROUPS = 1
;  893          1025  2      ELSE
;  894          1026  3          BEGIN
;  895          1027  3          GROUPS = 0;
;  896          1028  3          WHILE .STATUS
;  897          1029  3          DO
;  898          1030  4              BEGIN
;  899          1031  4              STR$TRIM (DSC, DSC, DSC [DSC$W_LENGTH] );
;  900          1032  4              STATUS = OTS$CVT_TI_L (DSC, BITVALUE);
;  901          1033  4              IF NOT .STATUS
;  902          1034  4              THEN
;  903          1035  5                  BEGIN
;  904          1036  5                  SIGNAL (LCP$_IVQUAL, 2, QUAL_GROUPS, DSC, .STATUS);
;  905          1037  5                  RETURN FAILURE
;  906          1038  5                  END
;  907          1039  4              ;
;  908          1040  4              IF
;  909          1041  4                  .BITVALUE LSS 0
;  910          1042  4                  OR
;  911          1043  4                  .BITVALUE GTR 255
;  912          1044  4              THEN
```

```
;  913          1045  5                 BEGIN
;  914          1046  5                 SIGNAL (LCP$_IVQUAL, 2, QUAL_GROUPS, DSC);
;  915          1047  5                 RETURN FAILURE
;  916          1048  5                 END
;  917          1049  4             ;
;  918          1050  4             GROUPS [0, .BITVALUE, 1, 0] = TRUE;
;  919          1051  4             $SETDSC (DSC, NUMLEN, NUMLST);
;  920          1052  4             STATUS = CLI$GET_VALUE (QUAL_GROUPS, DSC);
;  921          1053  4             END
;  922          1054  3         ;
;  923          1055  3         END
;  924          1056  2     ;
;  925          1057  2
;  926          1058  2     !
;  927          1059  2     ! The host node rating
;  928          1060  2     !
;  929          1061  2     $SETDSC (NUMLSTDSC, NUMLEN, NUMLST);
;  930          1062  2     IF CLI$GET_VALUE (QUAL_RATING, NUMLSTDSC)
;  931          1063  2     THEN
;  932          1064  3         BEGIN
;  933          1065  3         STR$TRIM (NUMLSTDSC, NUMLSTDSC, NUMLSTDSC [DSC$W_LENGTH] );
;  934          1066  3         STATUS = OTS$CVT_TI_L (NUMLSTDSC, RATING);
;  935          1067  3         IF NOT .STATUS
;  936          1068  3             OR
;  937          1069  4             (.RATING GEQU 255)
;  938          1070  3         THEN
;  939          1071  4             BEGIN
;  940          1072  4             SIGNAL (LCP$_IVQUAL, 2, QUAL_RATING, NUMLSTDSC, .STATUS);
;  941          1073  4             RETURN FAILURE
;  942          1074  4             END
;  943          1075  3         END
;  944          1076  2     ;
;  945          1077  2
;  946          1078  2     !
;  947          1079  2     ! The service name rating
;  948          1080  2     !
;  949          1081  2     $SETDSC (NUMLSTDSC, NUMLEN, NUMLST);
;  950          1082  2     IF CLI$GET_VALUE (QUAL_SERV_RATING, NUMLSTDSC)
;  951          1083  2     THEN
;  952          1084  3         BEGIN
;  953          1085  3         STR$TRIM (NUMLSTDSC, NUMLSTDSC, NUMLSTDSC [DSC$W_LENGTH] );
;  954          1086  3         STATUS = OTS$CVT_TI_L (NUMLSTDSC, SERVRATING);
;  955          1087  3         IF NOT .STATUS
;  956          1088  3             OR
;  957          1089  4             (.SERVRATING GEQU 255)
;  958          1090  3         THEN
;  959          1091  4             BEGIN
;  960          1092  4             SIGNAL (LCP$_IVQUAL, 2, QUAL_SERV_RATING, NUMLSTDSC, .STATUS);
;  961          1093  4             RETURN FAILURE
;  962          1094  4             END
;  963          1095  3         END
;  964          1096  2     ;
;  965          1097  2
;  966          1098  2     !
;  967          1099  2     ! Multicast timer setting
;  968          1100  2     !
;  969          1101  2     $SETDSC (NUMLSTDSC, NUMLEN, NUMLST);
```

```
: 970   1102  2     IF CLI$GET_VALUE (QUAL_MCAST, NUMLSTDSC)
: 971   1103  2     THEN
: 972   1104  3         BEGIN
: 973   1105  3         STR$TRIM (NUMLSTDSC, NUMLSTDSC, NUMLSTDSC [DSC$W_LENGTH] );
: 974   1106  3         STATUS = OTS$CVT_TI_L (NUMLSTDSC, MCASTIMR);
: 975   1107  3         IF NOT .STATUS
: 976   1108  3         THEN
: 977   1109  4             BEGIN
: 978   1110  4             SIGNAL (LCP$_IVQUAL, 2, QUAL_MCAST, NUMLSTDSC, .STATUS);
: 979   1111  4             RETURN FAILURE
: 980   1112  4             END
: 981   1113  3         ELSE
: 982   1114  4             IF  (.MCASTIMR GTRU 255)
: 983   1115  3                 OR
: 984   1116  4                 (.MCASTIMR LSS 10)
: 985   1117  3             THEN
: 986   1118  4                 BEGIN
: 987   1119  4                 SIGNAL (LCP$_IVQUAL, 2, QUAL_MCAST, NUMLSTDSC);
: 988   1120  4                 RETURN FAILURE
: 989   1121  4                 END
: 990   1122  3         END
: 991   1123  3
: 992   1124  2     ;
: 993   1125  2
: 994   1126  2     SUCCESS
: 995   1127  2
: 996   1128  1     END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE.2

                        45  44  4F  4E  24  53  59  53  00CAC P.AEV:  .ASCII  \SYS$NODE\                    ;
                                            010E0008  00CB4 P.AEU:  .LONG   17694728                        ;
                                            00000000' 00CB8         .ADDRESS P.AEV                          ;
            45  43  4E  55  4F  4E  4E  41  24  53  59  53  00CBC P.AEX:  .ASCII  \SYS$ANNOUNCE\              ;
                                            010E000C  00CC8 P.AEW:  .LONG   17694732                        ;
                                            00000000' 00CCC         .ADDRESS P.AEX                          ;
4D  41  4E  5F  45  43  49  56  52  45  53  24  53  59  53  00CD0 P.AEZ:  .ASCII  \SYS$SERVICE_NAME\          ;
                                                        45  00CDF                                           ;
                                            010E0010  00CE0 P.AEY:  .LONG   17694736                        ;
                                            00000000' 00CE4         .ADDRESS P.AEZ                          ;
            45  43  4E  55  4F  4E  4E  41  24  53  59  53  00CE8 P.AFB:  .ASCII  \SYS$ANNOUNCE\              ;
                                            010E000C  00CF4 P.AFA:  .LONG   17694732                        ;
                                            00000000' 00CF8         .ADDRESS P.AFB                          ;


                                                                .PSECT  $CODE$,NOWRT,2

                                        0FFC 00000 LCP_QUAL:
                                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11    ; 0932
                                5B 00000000G  00  9E 00002      MOVAB   LIB$SIGNAL, R11                         ;
                                5A 00000000G  8F  D0 00009      MOVL    #LCP$_IVQUAL, R10                       ;
                                59     0000V  CF  9E 00010      MOVAB   LCP_GETSTR, R9                          ;
                                58 00000000G  00  9E 00015      MOVAB   OTS$CVT_TI_L, R8                        ;
                                57 00000000G  00  9E 0001C      MOVAB   STR$TRIM, R7                            ;
```

```
                    56 00000000G  00  9E 00023          MOVAB   CLI$GET_VALUE, R6
                    55     0000'  CF  9E 0002A          MOVAB   QUAL_GROUPS, R5
                    54     0000'  CF  9E 0002F          MOVAB   NUMLSTDSC, R4
                    5E            0C  C2 00034          SUBL2   #12, SP
                    7E     10D9   8F  3C 00037          MOVZWL  #4313, -(SP)                    0975
                           0C54   C5  9F 0003C          PUSHAB  P.AEU
                           FF30   C4  9F 00040          PUSHAB  NAME
                                  10  DD 00044          PUSHL   #16
                           FF40   C4  9F 00046          PUSHAB  NAMEDSC
                             D0   A5  9F 0004A          PUSHAB  QUAL_NAME
                    69            06  FB 0004D          CALLS   #6, LCP_GETSTR
                    0C            50  E9 00050          BLBC    R0, 1$
                           FF40   C4  9F 00053          PUSHAB  NAMEDSC                         0978
             0000V  CF            01  FB 00057          CALLS   #1, LCP_FIXNAME
                    09            50  E8 0005C          BLBS    R0, 2$
                           FF40   C4  9F 0005F  1$:     PUSHAB  NAMEDSC                         0981
                             D0   A5  9F 00063          PUSHAB  QUAL_NAME
                                  43  11 00066          BRB     3$
                                  7E  D4 00068  2$:     CLRL    -(SP)                           0989
                           0C68   C5  9F 0006A          PUSHAB  P.AEW
                             A8   A4  9F 0006E          PUSHAB  NODE_ID
                    7E       40   8F  9A 00071          MOVZBL  #64, -(SP)
                             E8   A4  9F 00075          PUSHAB  NODE_IDDSC
                           0120   C5  9F 00078          PUSHAB  QUAL_NODEID
                    69            06  FB 0007C          CALLS   #6, LCP_GETSTR
                                  7E  D4 0007F          CLRL    -(SP)                           0995
                           0C80   C5  9F 00081          PUSHAB  P.AEY
                           FF48   C4  9F 00085          PUSHAB  SERV
                                  10  DD 00089          PUSHL   #16
                           FF58   C4  9F 0008B          PUSHAB  SERVDSC
                             3C   A5  9F 0008F          PUSHAB  QUAL_SERV_NAME
                    69            06  FB 00092          CALLS   #6, LCP_GETSTR
                    15            50  E9 00095          BLBC    R0, 4$
                           FF58   C4  9F 00098          PUSHAB  SERVDSC                         0999
             0000V  CF            01  FB 0009C          CALLS   #1, LCP_FIXNAME
                    09            50  E8 000A1          BLBS    R0, 4$
                           FF58   C4  9F 000A4          PUSHAB  SERVDSC                         1002
                             3C   A5  9F 000A8          PUSHAB  QUAL_SERV_NAME
                                  78  11 000AB  3$:     BRB     10$
                                  7E  D4 000AD  4$:     CLRL    -(SP)                           1011
                           0C94   C5  9F 000AF          PUSHAB  P.AFA
                           FF60   C4  9F 000B3          PUSHAB  ID
                    7E       40   8F  9A 000B7          MOVZBL  #64, -(SP)
                             A0   A4  9F 000BB          PUSHAB  IDDSC
                             F0   A5  9F 000BE          PUSHAB  QUAL_ID
                    69            06  FB 000C1          CALLS   #6, LCP_GETSTR
                 04 AE            10  D0 000C4          MOVL    #16, DSC                        1016
                 08 AE       F0   A4  9E 000C8          MOVAB   NUMLST, DSC+4
                             04   AE  9F 000CD          PUSHAB  DSC                             1017
                                  55  DD 000D0          PUSHL   R5
                    66            02  FB 000D2          CALLS   #2, CLI$GET_VALUE
                    53            50  D0 000D5          MOVL    R0, STATUS
                    06            53  E8 000D8          BLBS    STATUS, 5$                      1022
                 08 A4            01  D0 000DB          MOVL    #1, GROUPS                      1024
                                  62  11 000DF          BRB     13$
                             08   A4  D4 000E1  5$:     CLRL    GROUPS                          1027
                    5C            53  E9 000E4  6$:     BLBC    STATUS, 13$                     1028
```

```
                                    04   AE  9F 000E7          PUSHAB  DSC                                  ; 1031
                                    08   AE  9F 000EA          PUSHAB  DSC                                  ;
                                    0C   AE  9F 000ED          PUSHAB  DSC                                  ;
                            67           03  FB 000F0          CALLS   #3, STR$TRIM                         ;
                                         5E  DD 000F3          PUSHL   SP                                   ; 1032
                                    08   AE  9F 000F5          PUSHAB  DSC                                  ;
                            68           02  FB 000F8          CALLS   #2, OTS$CVT_TI_L                     ;
                            53           50  D0 000FB          MOVL    R0, STATUS                           ;
                            11           53  E8 000FE          BLBS    STATUS, 8$                           ; 1033
                                         53  DD 00101          PUSHL   STATUS                               ; 1036
                                    08   AE  9F 00103          PUSHAB  DSC                                  ;
                                         55  DD 00106          PUSHL   R5                                   ;
                                         02  DD 00108 7$:      PUSHL   #2                                   ;
                                         5A  DD 0010A          PUSHL   R10                                  ;
                            6B           05  FB 0010C          CALLS   #5, LIB$SIGNAL                       ;
                                       00FD  31 0010F          BRW     22$                                  ; 1037
                            52           6E  D0 00112 8$:      MOVL    BITVALUE, R2                         ; 1041
                                         09  19 00115          BLSS    9$                                   ;
                     000000FF   8F       52  D1 00117          CMPL    R2, #255                             ; 1043
                                         08  15 0011E          BLEQ    11$                                  ;
                                    04   AE  9F 00120 9$:      PUSHAB  DSC                                  ; 1046
                                         55  DD 00123          PUSHL   R5                                   ;
                                       00E0  31 00125 10$:     BRW     21$                                  ;
                 00         08   A4      52  E2 00128 11$:     BBSS    R2, GROUPS, 12$                      ; 1050
                            04   AE      10  D0 0012D 12$:     MOVL    #16, DSC                             ; 1051
                            08   AE  F0  A4  9E 00131          MOVAB   NUMLST, DSC+4                        ;
                                    04   AE  9F 00136          PUSHAB  DSC                                  ; 1052
                                         55  DD 00139          PUSHL   R5                                   ;
                            66           02  FB 0013B          CALLS   #2, CLI$GET_VALUE                    ;
                            53           50  D0 0013E          MOVL    R0, STATUS                           ;
                                         A1  11 00141          BRB     6$                                   ; 1028
                            64           10  D0 00143 13$:     MOVL    #16, NUMLSTDSC                       ; 1061
                            04   A4  F0  A4  9E 00146          MOVAB   NUMLST, NUMLSTDSC+4                  ;
                                         54  DD 0014B          PUSHL   R4                                   ; 1062
                                    10   A5  9F 0014D          PUSHAB  QUAL_RATING                          ;
                            66           02  FB 00150          CALLS   #2, CLI$GET_VALUE                    ;
                            2A           50  E9 00153          BLBC    R0, 16$                              ;
                                         54  DD 00156          PUSHL   R4                                   ; 1065
                                         54  DD 00158          PUSHL   R4                                   ;
                                         54  DD 0015A          PUSHL   R4                                   ;
                            67           03  FB 0015C          CALLS   #3, STR$TRIM                         ;
                                    28   A4  9F 0015F          PUSHAB  RATING                               ; 1066
                                         54  DD 00162          PUSHL   R4                                   ;
                            68           02  FB 00164          CALLS   #2, OTS$CVT_TI_L                     ;
                            53           50  D0 00167          MOVL    R0, STATUS                           ;
                            0A           53  E9 0016A          BLBC    STATUS, 14$                          ; 1067
                     000000FF   8F   28  A4  D1 0016D          CMPL    RATING, #255                         ; 1069
                                         09  1F 00175          BLSSU   16$                                  ;
                                         53  DD 00177 14$:     PUSHL   STATUS                               ; 1072
                                         54  DD 00179          PUSHL   R4                                   ;
                                    10   A5  9F 0017B          PUSHAB  QUAL_RATING                          ;
                                         88  11 0017E 15$:     BRB     7$                                   ;
                            64           10  D0 00180 16$:     MOVL    #16, NUMLSTDSC                       ; 1081
                            04   A4  F0  A4  9E 00183          MOVAB   NUMLST, NUMLSTDSC+4                  ;
                                         54  DD 00188          PUSHL   R4                                   ; 1082
                                    28   A5  9F 0018A          PUSHAB  QUAL_SERV_RATING                     ;
                            66           02  FB 0018D          CALLS   #2, CLI$GET_VALUE                    ;
```

```
                                 2A          50  E9 00190          BLBC    R0, 18$
                                             54  DD 00193          PUSHL   R4                                  1085
                                             54  DD 00195          PUSHL   R4
                                             54  DD 00197          PUSHL   R4
                                 67          03  FB 00199          CALLS   #3, STR$TRIM
                                        2C   A4  9F 0019C          PUSHAB  SERVRATING                          1086
                                             54  DD 0019F          PUSHL   R4
                                 68          02  FB 001A1          CALLS   #2, OTS$CVT_TI_L
                                 53          50  D0 001A4          MOVL    R0, STATUS
                                 0A          53  E9 001A7          BLBC    STATUS, 17$                         1087
                       000000FF  8F        2C   A4  D1 001AA          CMPL    SERVRATING, #255                    1089
                                             09  1F 001B2          BLSSU   18$
                                             53  DD 001B4 17$:      PUSHL   STATUS                              1092
                                             54  DD 001B6          PUSHL   R4
                                        28   A5  9F 001B8          PUSHAB  QUAL_SERV_RATING
                                             C1  11 001BB          BRB     15$
                                 64          10  D0 001BD 18$:      MOVL    #16, NUMLSTDSC                      1101
                             04  A4        F0   A4  9E 001C0          MOVAB   NUMLST, NUMLSTDSC+4
                                             54  DD 001C5          PUSHL   R4                                  1102
                                      00CC   C5  9F 001C7          PUSHAB  QUAL_MCAST
                                 66          02  FB 001CB          CALLS   #2, CLI$GET_VALUE
                                 41          50  E9 001CE          BLBC    R0, 23$
                                             54  DD 001D1          PUSHL   R4                                  1105
                                             54  DD 001D3          PUSHL   R4
                                             54  DD 001D5          PUSHL   R4
                                 67          03  FB 001D7          CALLS   #3, STR$TRIM
                                        38   A4  9F 001DA          PUSHAB  MCASTIMR                            1106
                                             54  DD 001DD          PUSHL   R4
                                 68          02  FB 001DF          CALLS   #2, OTS$CVT_TI_L
                                 53          50  D0 001E2          MOVL    R0, STATUS
                                 0A          53  E8 001E5          BLBS    STATUS, 19$                         1107
                                             53  DD 001E8          PUSHL   STATUS                              1110
                                             54  DD 001EA          PUSHL   R4
                                      00CC   C5  9F 001EC          PUSHAB  QUAL_MCAST
                                             8C  11 001F0          BRB     15$
                       000000FF  8F        38   A4  D1 001F2 19$:      CMPL    MCASTIMR, #255                      1114
                                             06  1A 001FA          BGTRU   20$
                                 0A        38   A4  D1 001FC          CMPL    MCASTIMR, #10                       1116
                                             10  18 00200          BGEQ    23$
                                             54  DD 00202 20$:      PUSHL   R4                                  1119
                                      00CC   C5  9F 00204          PUSHAB  QUAL_MCAST
                                             02  DD 00208 21$:      PUSHL   #2
                                             5A  DD 0020A          PUSHL   R10
                                 6B          04  FB 0020C          CALLS   #4, LIB$SIGNAL
                                             50  D4 0020F 22$:      CLRL    R0                                  1120
                                                 04 00211          RET
                                 50          01  D0 00212 23$      MOVL    #1, R0                              1128
                                                 04 00215          RET

; Routine Size:  534 bytes,    Routine Base:  $CODE$ + 00F0
```

```
%SBTTL  'LCP_GETSTR  Process a string value for a qualifier'
ROUTINE LCP_GETSTR (QUALDSC, DSC, SIZE, STR, LOGDSC, SYID) =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Accept a string value for a qualifier and translate it in the
!   case its a logical name. If there is not return value from the
!   call to CLI then use the LOGDSC to get our own interpretation.
!   (This should be done by the CLI, but it doesn't work that way.)
!
!   Also, if the LOGDSC doesn't translate, then attempt to use the
!   GETSYI parameter to retrieve a value.
!
! FORMAL PARAMETERS:
!
!   qualdsc             address of descriptor of the qualifier string
!   dsc                 descriptor to leave string
!   size                size of string
!   str                 string
!   logdsc              descriptor of logical name for defaulting
!   syid                system id parameter for $GETSYI call
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       success if sring is present, failure if not, other errors signalled
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LITERAL
        ITEM_SIZE = 16
        ;

    MACRO
        ITM_LEN = 0,0,16,0%,
        ITM_ID  = 2,0,16,0%,
        ITM_RETBUF = 4,0,32,0%,
        ITM_RETLEN = 8,0,32,0%,
        ITM_END = 12,0,32,0%
        ;

    MAP
```

```
; 1055          1186  2          STR : REF VECTOR [,BYTE],
; 1056          1187  2          DSC : REF BLOCK [,BYTE],
; 1057          1188  2          LOGDSC : REF BLOCK [,BYTE],
; 1058          1189  2          SYID
; 1059          1190  2          ;
; 1060          1191  2
; 1061          1192  2      LOCAL
; 1062          1193  2          RDSC     : BLOCK [DSC$K_Z_BLN, BYTE],
; 1063          1194  2          PTR      : REF VECTOR [,BYTE],
; 1064          1195  2          ITMLST   : BLOCK [ITEM_SIZE, BYTE],
; 1065          1196  2          IOSB     : VECTOR [4, WORD],
; 1066          1197  2          RET_LEN,
; 1067          1198  2          LEN,
; 1068          1199  2          STATUS
; 1069          1200  2          ;
; 1070          1201  2
; 1071          1202  2      !
; 1072          1203  2      ! Obtain the string
; 1073          1204  2      !
; 1074          1205  2      $SETDSC (DSC, .SIZE, .STR);
; 1075          1206  2      STATUS = CLI$GET_VALUE (.QUALDSC, .DSC);
; 1076          1207  2      IF NOT .STATUS
; 1077          1208  2      THEN
; 1078          1209  3          BEGIN
; 1079          1210  3          $SETDSC (DSC, .SIZE, .STR);              ! Re-set descriptor
; 1080          1211  3          !
; 1081          1212  3          !  If there is no logical name or translation OR no system parameter,
; 1082          1213  3          !  then return failure.
; 1083          1214  3          !
; 1084          1215  4          IF NOT  (.LOGDSC NEQ 0
; 1085          1216  4                  AND
; 1086        P 1217  5                  (STATUS = $TRNLOG
; 1087        P 1218  5                      (LOGNAM = .LOGDSC, RSLLEN = DSC[DSC$W_LENGTH],
; 1088          1219  4                      RSLBUF = .DSC)))
; 1089          1220  3              OR
; 1090          1221  3              .STATUS EQL SS$_NOTRAN
; 1091          1222  3          THEN
; 1092          1223  3              IF .SYID NEQ 0
; 1093          1224  3              THEN
; 1094          1225  4                  BEGIN
; 1095          1226  4                  ITMLST [ITM_LEN] = .SIZE;
; 1096          1227  4                  ITMLST [ITM_ID] = .SYID;
; 1097          1228  4                  ITMLST [ITM_RETBUF] = .STR;
; 1098          1229  4                  ITMLST [ITM_RETLEN] = DSC [DSC$W_LENGTH];
; 1099          1230  4                  ITMLST [ITM_END] = 0;    ! End of LIST.
; 1100        P 1231  5                  IF NOT (STATUS = $GETSYIW ( ITMLST = ITMLST,
; 1101          1232  6                                  IOSB = IOSB)
; 1102          1233  5                          OR
; 1103          1234  5                          .IOSB [0] )
; 1104          1235  4                  THEN
; 1105          1236  5                      BEGIN
; 1106          1237  5                      DSC [DSC$W_LENGTH] = 0;
; 1107          1238  5                      RETURN FAILURE
; 1108          1239  5                      END
; 1109          1240  4                  END
; 1110          1241  3              ELSE
; 1111          1242  4                  BEGIN
```

```
; 1112          1243  4                 DSC [DSC$W_LENGTH] = 0;
; 1113          1244  4                 RETURN FAILURE
; 1114          1245  4                 END
; 1115          1246  3             ;
; 1116          1247  3         END
; 1117          1248  2     ;
; 1118          1249  2
; 1119          1250  2     !
; 1120          1251  2     ! Translate the name in case its a logical name
; 1121          1252  2     !
; 1122          1253  2     $SETDSC (RDSC, .SIZE, .STR);
; 1123          1254  2     DECR DCX FROM 10 TO 1
; 1124          1255  2     DO
; 1125          1256  3         BEGIN
; 1126          1257  3         STR$TRIM (.DSC, .DSC, DSC [DSC$W_LENGTH] );
; 1127        P 1258  3         STATUS = $TRNLOG
; 1128          1259  3             (LOGNAM = .DSC, RSLLEN = RDSC [DSC$W_LENGTH], RSLBUF = RDSC);
; 1129          1260  3         IF .STATUS EQL SS$_NOTRAN
; 1130          1261  3         THEN
; 1131          1262  3             EXITLOOP
; 1132          1263  3         ;
; 1133          1264  3         IF NOT .STATUS
; 1134          1265  3         THEN
; 1135          1266  4             BEGIN
; 1136          1267  4             SIGNAL (LCP$_IVQUAL, 2, .QUALDSC, .DSC, .STATUS);
; 1137          1268  4             RETURN FAILURE
; 1138          1269  4             END
; 1139          1270  3         END
; 1140          1271  2     ;
; 1141          1272  2
; 1142          1273  2     !
; 1143          1274  2     ! Trim off the trailing spaces and look for an indirect or untranslated
; 1144          1275  2     ! logical name.  If either, then just return none available.
; 1145          1276  2     !
; 1146          1277  2     STR$TRIM (.DSC, .DSC, DSC [DSC$W_LENGTH] );
; 1147          1278  2     PTR = .DSC [DSC$A_POINTER];
; 1148          1279  2     LEN = .DSC [DSC$W_LENGTH];
; 1149          1280  2
; 1150          1281  2     !
; 1151          1282  2     ! Scan the name and translate the non-printables to spaces
; 1152          1283  2     !
; 1153          1284  2     DECR DCX FROM .LEN - 1 TO 0
; 1154          1285  2     DO
; 1155          1286  3         BEGIN
; 1156          1287  3         SELECTONE TRUE OF
; 1157          1288  3         SET
; 1158          1289  3         [.PTR [.DCX] LSS ' '] : PTR [.DCX] = ' ';
; 1159          1290  3         [OTHERWISE] : 0;
; 1160          1291  3         TES
; 1161          1292  3         END
; 1162          1293  2     ;
; 1163          1294  2
; 1164          1295  2     DECR DCX FROM .LEN TO 0
; 1165          1296  2     DO
; 1166          1297  3         BEGIN
; 1167          1298  3         IF .PTR [0] EQL ' '
; 1168          1299  3         THEN
```

```
; 1169   1300  4            BEGIN
; 1170   1301  4            PTR = DSC [DSC$A_POINTER] = .PTR + 1;
; 1171   1302  4            LEN = DSC [DSC$W_LENGTH] = .LEN - 1
; 1172   1303  4            END
; 1173   1304  3        END
; 1174   1305  2    ;
; 1175   1306  2    IF .PTR [0] EQL '@'
; 1176   1307  2        OR
; 1177   1308  2        CH$EQL (MIN (4, .LEN), .PTR, 4, UPLIT ('SYS$') )
; 1178   1309  2    THEN
; 1179   1310  3        BEGIN
; 1180   1311  3        DSC [DSC$W_LENGTH] = MIN (.QUAL_NONE [DSC$W_LENGTH], .SIZE);
; 1181   1312  3        DSC [DSC$A_POINTER] = .QUAL_NONE [DSC$A_POINTER];
; 1182   1313  3        RETURN FAILURE
; 1183   1314  3        END
; 1184   1315  2    ;
; 1185   1316  2
; 1186   1317  2    SUCCESS
; 1187   1318  2
; 1188   1319  1    END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2
                                24  53  59  53  00CFC P.AFC:  .ASCII  \SYS$\                                    ;
                                                        .EXTRN  SYS$TRNLOG, SYS$GETSYIW
                                                        .PSECT  $CODE$,NOWRT,2
                                        01FC 00000 LCP_GETSTR:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8                      ; 1130
                            58 00000000G  0/  9E 00002  MOVAB   STR$TRIM, R8
                            57 00000000G  C0  9E 00009  MOVAB   SYS$TRNLOG, R7
                            5E            20  C2 00010  SUBL2   #32, SP
                            54       08   AC  7D 00013  MOVQ    DSC, R4                                       ; 1205
                            64            55  3C 00017  MOVZWL  R5, (R4)
                            56       04   A4  9E 0001A  MOVAB   4(R4), R6
                            66       10   AC  D0 0001E  MOVL    STR, (R6)
                                          54  DD 00022  PUSHL   R4                                            ; 1206
                                     04   AC  DD 00024  PUSHL   QUALDSC
                     00000000G      00    02  FB 00027  CALLS   #2, CLI$GET_VALUE
                            52            50  D0 0002E  MOVL    R0, STATUS
                            63            52  E8 00031  BLBS    STATUS, 3$                                    ; 1207
                            64            55  3C 00034  MOVZWL  R5, (R4)                                      ; 1210
                            66       10   AC  D0 00037  MOVL    STR, (R6)
                                     14   AC  D5 0003B  TSTL    LOGDSC                                        ; 1215
                                          1D  13 0003E  BEQL    1$
                                          7E  7C 00040  CLRQ    -(SP)                                         ; 1219
                                          7E  D4 00042  CLRL    -(SP)
                                          54  DD 00044  PUSHL   R4
                                          54  DD 00046  PUSHL   R4
                                     14   AC  DD 00048  PUSHL   LOGDSC
                            67            06  FB 0004B  CALLS   #6, SYS$TRNLOG
                            52            50  D0 0004E  MOVL    R0, STATUS
                            09            52  E9 00051  BLBC    STATUS, 1$
```

```
                00000629  8F          52  D1 00054          CMPL    STATUS, #1577                          ; 1221
                                      3A  12 0005B          BNEQ    3$                                     ;
                                  18  AC  D5 0005D 1$:      TSTL    SYID                                   ; 1223
                                      31  13 00060          BEQL    2$                                     ;
                      08  AE          55  B0 00062          MOVW    R5, ITMLST                             ; 1226
                      0A  AE      18  AC  B0 00066          MOVW    SYID, ITMLST+2                         ; 1227
                      0C  AE      10  AC  D0 0006B          MOVL    STR, ITMLST+4                          ; 1228
                      10  AE          54  D0 00070          MOVL    R4, ITMLST+8                           ; 1229
                                  14  AE  D4 00074          CLRL    ITMLST+12                              ; 1230
                                      7E  7C 00077          CLRQ    -(SP)                                  ; 1232
                                  08  AE  9F 00079          PUSHAB  IOSB                                   ;
                                  14  AE  9F 0007C          PUSHAB  ITMLST                                 ;
                                      7E  7C 0007F          CLRQ    -(SP)                                  ;
                                      7E  D4 00081          CLRL    -(SP)                                  ;
                00000000G  00       07  FB 00083          CALLS   #7, SYS$GETSYIW                        ;
                           52       6E  3C 0008A          MOVZWL  IOSB, STATUS                           ; 1234
                           52       50  C8 0008D          BISL2   R0, STATUS                             ;
                           04       52  E8 00090          BLBS    STATUS, 3$                             ; 1233
                                    64  B4 00093 2$:      CLRW    (R4)                                   ; 1243
                                    49  11 00095          BRB     5$                                     ; 1244
                      18  AE        55  3C 00097 3$:      MOVZWL  R5, RDSC                               ; 1253
                      1C  AE      10  AC  D0 0009B          MOVL    STR, RDSC+4                            ;
                           53       0A  D0 000A0          MOVL    #10, DCX                               ; 1254
                                    54  DD 000A3 4$:      PUSHL   R4                                     ; 1257
                                    54  DD 000A5          PUSHL   R4                                     ;
                                    54  DD 000A7          PUSHL   R4                                     ;
                           68       03  FB 000A9          CALLS   #3, STR$TRIM                           ;
                                    7E  7C 000AC          CLRQ    -(SP)                                  ; 1259
                                    7E  D4 000AE          CLRL    -(SP)                                  ;
                                  24  AE  9F 000B0          PUSHAB  RDSC                                   ;
                                  28  AE  9F 000B3          PUSHAB  RDSC                                   ;
                                    54  DD 000B6          PUSHL   R4                                     ;
                           67       06  FB 000B8          CALLS   #6, SYS$TRNLOG                         ;
                           52       50  D0 000BB          MOVL    R0, STATUS                             ;
                00000629   8F       52  D1 000BE          CMPL    STATUS, #1577                          ; 1260
                                    1E  13 000C5          BEQL    7$                                     ;
                           18       52  E8 000C7          BLBS    STATUS, 6$                             ; 1264
                                    52  DD 000CA          PUSHL   STATUS                                 ; 1267
                                    54  DD 000CC          PUSHL   R4                                     ;
                                  04  AC  DD 000CE          PUSHL   QUALDSC                                ;
                                    02  DD 000D1          PUSHL   #2                                     ;
                           00000000G 8F DD 000D3          PUSHL   #LCP$_IVQUAL                           ;
                00000000G  00       05  FB 000D9          CALLS   #5, LIB$SIGNAL                         ;
                                    73  11 000E0 5$:      BRB     15$                                    ; 1268
                           BE       53  F5 000E2 6$:      SOBGTR  DCX, 4$                                ; 1264
                                    54  DD 000E5 7$:      PUSHL   R4                                     ; 1277
                                    54  DD 000E7          PUSHL   P4                                     ;
                                    54  DD 000E9          PUSHL   R4                                     ;
                           68       03  FB 000EB          CALLS   #3, STR$TRIM                           ;
                           50       66  D0 000EE          MOVL    (R6), PTR                              ; 1278
                           52       64  3C 000F1          MOVZWL  (R4), LEN                              ; 1279
                           51       52  D0 000F4          MOVL    LEN, DCX                               ; 1289
                                    0A  11 000F7          BRB     9$                                     ;
                           20     6140  91 000F9 8$:      CMPB    (DCX)[PTR], #32                        ;
                                    04  1E 000FD          BGEQU   9$                                     ;
                         6140       20  90 000FF          MOVB    #32, (DCX)[PTR]                        ;
                           F3       51  F4 00103 9$:      SOBGEQ  DCX, 8$                                ; 1286
```

```
                                51      01   A2  9E 00106          MOVAB   1(R2), DCX                          ; 1295
                                             19  11 0010A          BRB     11$                                ;
                                20           60  91 0010C 10$:    CMPB    (PTR), #32                         ; 1298
                                             14  12 0010F          BNEQ    11$                                ;
                                53      01   A0  9E 00111          MOVAB   1(R0), R3                          ; 1301
                                66           53  D0 00115          MOVL    R3, (R6)                           ;
                                50           53  D0 00118          MOVL    R3, PTR                            ;
                                53      FF   A2  9E 0011B          MOVAB   -1(R2), R3                         ; 1302
                                64           53  B0 0011F          MOVW    R3, (R4)                           ;
                                52           53  D0 00122          MOVL    R3, LEN                            ;
                                E4           51  F4 00125 11$:     SOBGEQ  DCX, 10$                           ; 1297
                        40      8F           60  91 00128          CMPB    (PTR), #64                         ; 1306
                                             12  13 0012C          BEQL    13$                                ;
                                04           52  D1 0012E          CMPL    R2, #4                             ; 1308
                                             03  15 00131          BLEQ    12$                                ;
                                52           04  D0 00133          MOVL    #4, R2                             ;
        04              00      60           52  2D 00136 12$:     CMPC5   R2, (PTR), #0, #4, P.AFC           ;
                                        0000'  CF     0013B                                                   ;
                                             18  12 0013E          BNEQ    16$                                ;
                                50      0000'  CF  3C 00140 13$:    MOVZWL  QUAL_NONE, R0                      ; 1311
                                55           50  D1 00145          CMPL    R0, R5                             ;
                                             03  15 00148          BLEQ    14$                                ;
                                50           55  D0 0014A          MOVL    R5, R0                             ;
                                64           50  B0 0014D 14$:     MOVW    R0, (R4)                           ;
                                66      0000'  CF  D0 00150          MOVL    QUAL_NONE+4, (R6)                  ; 1312
                                             50  D4 00155 15$:     CLRL    R0                                 ; 1313
                                                 04 00157          RET                                        ;
                                50           01  D0 00158 16$:     MOVL    #1, R0                             ; 1319
                                                 04 0015B          RET                                        ;

; Routine Size:  348 bytes,    Routine Base:  $CODE$ + 0306
```

```
%SBTTL  'LCP_FIXNAME  Process a name string'
ROUTINE LCP_FIXNAME (DSC) =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Process a name string.  Convert it to the correct form.
!
! FORMAL PARAMETERS:
!
!   DSC         Address of descriptor of string
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       success or failure
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    MAP
        DSC : REF BLOCK [,BYTE]
        ;

    LOCAL
        LEN,
        PTR : REF VECTOR [,BYTE],
        STATUS
        ;

    !
    ! Strip beginning underscores and trailing colons
    !
    PTR = .DSC [DSC$A_POINTER];
    LEN = .DSC [DSC$W_LENGTH];
    DO
        BEGIN
        STATUS = FALSE;
        IF .PTR [0] EQL '_'
        THEN
            BEGIN
            LEN = DSC [DSC$W_LENGTH] = .LEN - 1;
            PTR = DSC [DSC$A_POINTER] = .PTR + 1;
            STATUS = TRUE
```

```
; 1247   1377  4            END
; 1248   1378  3        ;
; 1249   1379  3        IF .PTR [.LEN - 1] EQL ':'
; 1250   1380  3        THEN
; 1251   1381  4            BEGIN
; 1252   1382  4            LEN = DSC [DSC$W_LENGTH] = .LEN - 1;
; 1253   1383  4            STATUS = TRUE
; 1254   1384  4            END
; 1255   1385  3        END
; 1256   1386  2    WHILE .STATUS
; 1257   1387  2    ;
; 1258   1388  2    !
; 1259   1389  2    ! Upcase the string and check for legal characters
; 1260   1390  2    !
; 1261   1391  2    STR$UPCASE (.DSC, .DSC);
; 1262   1392  2    DECR DCX FROM .LEN - 1 TO 0
; 1263   1393  2    DO
; 1264   1394  3        BEGIN
; 1265   1395  3        SELECTONE TRUE OF
; 1266   1396  3        SET
; 1267   1397  3        [.PTR [.DCX] GEQ 'A' AND .PTR [.DCX] LEQ 'Z'] : 0;
; 1268   1398  3        [.PTR [.DCX] GEQ '0' AND .PTR [.DCX] LEQ '9'] : 0;
; 1269   1399  3        [.PTR [.DCX] EQL ':'] : 0;
; 1270   1400  3        [.PTR [.DCX] EQL '$'] : 0;
; 1271   1401  3        [OTHERWISE] : RETURN FAILURE;
; 1272   1402  3        TES
; 1273   1403  3        END
; 1274   1404  2    ;
; 1275   1405  2
; 1276   1406  2    SUCCESS
; 1277   1407  2
; 1278   1408  1    END;
```

```
                                 003C 00000 LCP_FIXNAME:
                                                    .WORD   Save R2,R3,R4,R5                    ; 1321
        50      04   AC  D0 00002                   MOVL    DSC, R0                             ; 1366
        53      04   A0  D0 00006                   MOVL    4(R0), PTR
        52           60  3C 0000A                   MOVZWL  (R0), LEN                           ; 1367
                     54  D4 0000D 1$:               CLRL    STATUS                              ; 1370
    5F  8F           63  91 0000F                   CMPB    (PTR), #95                          ; 1371
                     18  12 00013                   BNEQ    2$
        51      FF   A2  9E 00015                   MOVAB   -1(R2), R1                          ; 1374
        60           51  B0 00019                   MOVW    R1, (R0)
        52           51  D0 0001C                   MOVL    R1, LEN
        51      01   A3  9E 0001F                   MOVAB   1(R3), R1                           ; 1375
    04  A0           51  D0 00023                   MOVL    R1, 4(R0)
        53           51  D0 00027                   MOVL    R1, PTR
        54           01  D0 0002A                   MOVL    #1, STATUS                          ; 1376
        3A      FF A243  91 0002D 2$:               CMPB    -1(LEN)[PTR], #58                   ; 1379
                     0D  12 00032                   BNEQ    3$
        51      FF   A2  9E 00034                   MOVAB   -1(R2), R1                          ; 1382
        60           51  B0 00038                   MOVW    R1, (R0)
        52           51  D0 0003B                   MOVL    R1, LEN
```

```
                                54          01  D0 0003E          MOVL    #1, STATUS                          ; 1383
                                C9          54  E8 00041 3$:      BLBS    STATUS, 1$                          ; 1386
                                            50  DD 00044          PUSHL   R0                                  ; 1391
                                            50  DD 00046          PUSHL   R0                                  ;
                     00000000G  00          02  FB 00048          CALLS   #2, STR$UPCASE                      ;
                                            4E  11 0004F          BRB     9$                                  ; 1392
                                51        6243  9A 00051 4$:      MOVZBL  (DCX)[PTR], R1                      ; 1397
                                            54  D4 00055          CLRL    R4                                  ;
                           41   8F          51  91 00057          CMPB    R1, #65                             ;
                                            02  1F 0005B          BLSSU   5$                                  ;
                                            54  D6 0005D          INCL    R4                                  ;
                                            50  D4 0005F 5$:      CLRL    R0                                  ;
                           5A   8F          51  91 00061          CMPB    R1, #90                             ;
                                            02  1A 00065          BGTRU   6$                                  ;
                                            50  D6 00067          INCL    R0                                  ;
                                55          54  D2 00069 6$:      MCOML   R4, R5                              ;
                                50          55  CA 0006C          BICL2   R5, R0                              ;
                                01          50  D1 0006F          CMPL    R0, #1                              ;
                                            2B  13 00072          BEQL    9$                                  ;
                                            54  D4 00074          CLRL    R4                                  ; 1398
                                30          51  91 00076          CMPB    R1, #48                             ;
                                            02  1F 00079          BLSSU   7$                                  ;
                                            54  D6 0007B          INCL    R4                                  ;
                                            50  D4 0007D 7$:      CLRL    R0                                  ;
                                39          51  91 0007F          CMPB    R1, #57                             ;
                                            02  1A 00082          BGTRU   8$                                  ;
                                            50  D6 00084          INCL    R0                                  ;
                                55          54  D2 00086 8$:      MCOML   R4, R5                              ;
                                50          55  CA 00089          BICL2   R5, R0                              ;
                                01          50  D1 0008C          CMPL    R0, #1                              ;
                                            0E  13 0008F          BEQL    9$                                  ;
                           5F   8F          51  91 00091          CMPB    R1, #95                             ; 1399
                                            08  13 00095          BEQL    9$                                  ;
                                24          51  91 00097          CMPB    R1, #36                             ; 1400
                                            03  13 0009A          BEQL    9$                                  ;
                                            50  D4 0009C          CLRL    R0                                  ; 1401
                                            04  0009E             RET                                         ;
                                AF          52  F4 0009F 9$:      SOBGEQ  DCX, 4$                             ; 1394
                                50          01  D0 000A2          MOVL    #1, R0                              ; 1408
                                            04  000A5             RET                                         ;

; Routine Size:  166 bytes,    Routine Base:  $CODE$ + 0462
```

```
; 1280   1409  1 %SBTTL  'LCP_START  Start the terminal port driver'
; 1281   1410  1 GLOBAL ROUTINE LCP_START : NOVALUE =
; 1282   1411  1
; 1283   1412  1 !++
; 1284   1413  1 ! FUNCTIONAL DESCRIPTION:
; 1285   1414  1 !
; 1286   1415  1 !   Handle the general START request and the START HISTORY request.
; 1287   1416  1 !   If it is the general start request, then set the multicast message
; 1288   1417  1 !   data and then start the port driver. If it's the START HISTORY
; 1289   1418  1 !   request, allocate the history buffer.
; 1290   1419  1 !
; 1291   1420  1 ! FORMAL PARAMETERS:
; 1292   1421  1 !
; 1293   1422  1 !       NONE
; 1294   1423  1 !
; 1295   1424  1 ! IMPLICIT INPUTS:
; 1296   1425  1 !
; 1297   1426  1 !       results from qualifier parse and address of driver comm area
; 1298   1427  1 !
; 1299   1428  1 ! IMPLICIT OUTPUTS:
; 1300   1429  1 !
; 1301   1430  1 !       NONE
; 1302   1431  1 !
; 1303   1432  1 ! ROUTINE VALUE:
; 1304   1433  1 ! COMPLETION CODES:
; 1305   1434  1 !
; 1306   1435  1 !       NONE
; 1307   1436  1 !
; 1308   1437  1 ! SIDE EFFECTS:
; 1309   1438  1 !
; 1310   1439  1 !       driver started
; 1311   1440  1 !
; 1312   1441  1 !--
; 1313   1442  1
; 1314   1443  2     BEGIN
; 1315   1444  2
; 1316   1445  2     LOCAL
; 1317   1446  2         STATUS
; 1318   1447  2         ;
; 1319   1448  2
; 1320   1449  2     !
; 1321   1450  2     !  Check if this is a START HISTORY command
; 1322   1451  2     !
; 1323   1452  2     $SETDSC (SUBCMDDSC, CMDLEN, SUBCMD);
; 1324   1453  2     STATUS = CLI$GET_VALUE (QUAL_SUBCOMMAND, SUBCMDDSC);
; 1325   1454  2     IF .STATUS EQL CLI$_ABSENT
; 1326   1455  2     THEN
; 1327   1456  3         BEGIN
; 1328   1457  3         !
; 1329   1458  3         !  Get the qualifiers and set them
; 1330   1459  3         !
; 1331   1460  4         IF NOT (STATUS = LCP_SET () )
; 1332   1461  3         THEN
; 1333   1462  3             RETURN
; 1334   1463  3         ;
; 1335   1464  3
; 1336   1465  3         !
```

```
; 1337   1466  3          !  Now start the port driver
; 1338   1467  3          !
; 1339   1468  4          IF (STATUS = $CMKRNL (ROUTIN = LCPK_START) )
; 1340   1469  3          THEN
; 1341   1470  4              LOG_SIGNAL (LCP$_STARTED)
; 1342   1471  3          ELSE
; 1343   1472  3              SIGNAL (LCP$_NOTSTARTED, 0, .STATUS)
; 1344   1473  3          ;
; 1345   1474  3          END
; 1346   1475  2      ELSE
; 1347   1476  3          BEGIN
; 1348   1477  3          STR$TRIM (SUBCMDDSC, SUBCMDDSC, SUBCMDDSC [DSC$W_LENGTH] );
; 1349   1478  3          !
; 1350   1479  3          ! Dispatch on the subcommand string
; 1351   1480  3          !
; 1352   1481  3          SELECTONE TRUE OF
; 1353   1482  3          SET
; 1354   1483  3              [ $CMPDSC (SUBCMDDSC, QUAL_HISTORY) ] :
; 1355   1484  4              (
; 1356   1485  5              IF NOT (STATUS = $CMKRNL (ROUTIN = LCPK_STARTHISTORY) )
; 1357   1486  4              THEN
; 1358   1487  4                  SIGNAL (LCP$_NOSTARTHIST, 0, .STATUS)
; 1359   1488  4              ;
; 1360   1489  3              );
; 1361   1490  3
; 1362   1491  3          [OTHERWISE] : SIGNAL (LCP$_IVCMD);
; 1363   1492  3
; 1364   1493  3          TES;
; 1365   1494  3
; 1366   1495  3          END
; 1367   1496  2      ;
; 1368   1497  1      END;


                                    03FC 00000          .ENTRY  LCP_START, Save R2,R3,R4,R5,R6,R7,R8,R9     ; 1410
                 59 00000000G  00    9E 00002          MOVAB   LIB$SIGNAL, R9                              ;
                 58 00000000G  00    9E 00009          MOVAB   SYS$CMKRNL, R8                              ;
                 57     0000'  CF    9E 00010          MOVAB   QUAL_HISTORY, R7                            ;
                 56     0000'  CF    9E 00015          MOVAB   SUBCMDDSC, R6                               ;
                 66            10    D0 0001A          MOVL    #16, SUBCMDDSC                              ; 1452
          04     A6       F0  A6    9E 0001D          MOVAB   SUBCMD, SUBCMDDSC+4                         ;
                               56    DD 00022          PUSHL   R6                                          ; 1453
                        FF20  C7    9F 00024          PUSHAB  QUAL_SUBCOMMAND                             ;
   00000000G     00            02    FB 00028          CALLS   #2, CLI$GET_VALUE                           ;
                 55            50    D0 0002F          MOVL    R0, STATUS                                  ;
   00000000G     8F            55    D1 00032          CMPL    STATUS, #CLI$_ABSENT                        ; 1454
                               3C    12 00039          BNEQ    3$                                          ;
       0000V     CF            00    FB 0003B          CALLS   #0, LCP_SET                                 ; 1460
                 55            50    D0 00040          MOVL    R0, STATUS                                  ;
                 01            55    E8 00043          BLBS    STATUS, 1$                                  ;
                                     04 00046          RET                                                 ;
                               7E    D4 00047 1$:      CLRL    -(SP)                                       ; 1468
                        0000V  CF    9F 00049          PUSHAB  LCPK_START                                  ;
                 68            02    FB 0004D          CALLS   #2, SYS$CMKRNL                              ;
```

```
                                55          50  D0 00050          MOVL    R0, STATUS
                                15          55  E9 00053          BLBC    STATUS, 2$
                                        68  A7  9F 00056          PUSHAB  QUAL_LOG                                    1470
                     00000000G  00          01  FB 00059          CALLS   #1, CLI$PRESENT
                                6D          50  E9 00060          BLBC    R0, 10$
                                   00000000G 8F  DD 00063          PUSHL   #LCP$_STARTED
                                             62  11 00069          BRB     9$
                                             55  DD 0006B 2$:      PUSHL   STATUS                                      1472
                                             7E  D4 0006D          CLRL    -(SP)
                                   00000000G 8F  DD 0006F          PUSHL   #LCP$_NOTSTARTED
                                             4C  11 00075          BRB     7$
                                             56  DD 00077 3$:      PUSHL   R6                                          1477
                                             56  DD 00079          PUSHL   R6
                                             56  DD 0007B          PUSHL   R6
                     00000000G  00          03  FB 0007D          CALLS   #3, STR$TRIM
                                50          66  3C 00084          MOVZWL  SUBCMDDSC, R0                               1483
                                50          67  B1 00087          CMPW    QUAL_HISTORY, R0
                                            03  1E 0008A          BGEQU   4$
                                50          67  3C 0008C          MOVZWL  QUAL_HISTORY, R0
                                03          50  B1 0008F 4$:      CMPW    R0, #3
                                            03  1E 00092          BGEQU   5$
                                50          03  D0 00094          MOVL    #3, R0
                                            54  D4 00097 5$:      CLRL    R4
     50         20      04     B6          66  2D 00099          CMPC5   SUBCMDDSC, @SUBCMDDSC+4, #32, R0, -
                                        04  B7     0009F                  @QUAL_HISTORY+4
                                            02  12 000A1          BNEQ    6$
                                            54  D6 000A3          INCL    R4
                                01          54  D1 000A5 6$:      CMPL    R4, #1
                                            1D  12 000A8          BNEQ    8$
                                            7E  D4 000AA          CLRL    -(SP)                                       1485
                                     0000V  CF  9F 000AC          PUSHAB  LCPK_STARTHISTORY
                                68          02  FB 000B0          CALLS   #2, SYS$CMKRNL
                                55          50  D0 000B3          MOVL    R0, STATUS
                                17          55  E8 000B6          BLBS    STATUS, 10$
                                            55  DD 000B9          PUSHL   STATUS                                      1487
                                            7E  D4 000BB          CLRL    -(SP)
                                   00000000G 8F  DD 000BD          PUSHL   #LCP$_NOSTARTHIST
                                69          03  FB 000C3 7$:      CALLS   #3, LIB$SIGNAL
                                                04 000C6          RET                                                 1481
                                   00000000G 8F  DD 000C7 8$:      PUSHL   #LCP$_IVCMD                                 1491
                                69          01  FB 000CD 9$:      CALLS   #1, LIB$SIGNAL
                                                04 000D0 10$:     RET                                                 1497

; Routine Size:  209 bytes,    Routine Base:  $CODE$ + 0508
```

```
1 %SBTTL  'LCP_STOP  Stop the port driver'
1 GLOBAL ROUTINE LCP_STOP : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !   The kernel routine is called to stop the driver.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       the driver is stopped
1 !
1 !--
1
2     BEGIN
2
2     LITERAL
2         STRSIZE = 32
2         ;
2
2     BIND
2         TTNAME = DESCRIPTOR ('TT')
2         ;
2
2     LOCAL
2         DSC : BLOCK [DSC$K_Z_BLN, BYTE],
2         STR : VECTOR [STRSIZE, BYTE],
2         STATUS
2         ;
2
2     !
2     !  Check if this is a STOP HISTORY command
2     !
2     $SETDSC (SUBCMDDSC, CMDLEN, SUBCMD);
2     STATUS = CLI$GET_VALUE (QUAL_SUBCOMMAND, SUBCMDDSC);
2     IF .STATUS EQL CLI$_ABSENT
2     THEN
2         BEGIN
3     !
3     !  This is a general STOP command
```

```
1427          1555  3      !
1428          1556  3          !
1429          1557  3          !  We cannot stop LAT from a LAT terminal
1430          1558  3          !
1431          1559  3          $SETDSC (DSC, STRSIZE, STR);
1432        P 1560  3          $TRNLOG
1433        P 1561  3              (
1434        P 1562  3              LOGNAM = TTNAME,
1435        P 1563  3              RSLLEN = DSC [DSC$W_LENGTH],
1436        P 1564  3              RSLBUF = DSC
1437          1565  3              );
1438          1566  3          IF CH$EQL (4, UPLIT (%ASCII '_LTA'), 4, STR)
1439          1567  3              OR
1440          1568  4             ($CMKRNL (ROUTIN = LCPK_IS_LAT_TERMINAL) )
1441          1569  3          THEN
1442          1570  4              BEGIN
1443          1571  4              SIGNAL (LCP$_NOTFROMLAT);
1444          1572  4              RETURN
1445          1573  4              END
1446          1574  3          ;
1447          1575  3
1448          1576  3          $CMKRNL (ROUTIN = LCPK_STOPHISTORY);    ! Delete the history buffer
1449          1577  3
1450          1578  3          !
1451          1579  3          !  Stop the LTDRIVER
1452          1580  3          !
1453          1581  4          IF (STATUS = $CMKRNL (ROUTIN = LCPK_STOP) )
1454          1582  3          THEN
1455          1583  4              LOG_SIGNAL (LCP$_STOPPED)
1456          1584  3          ELSE
1457          1585  3              SIGNAL (LCP$_NOTSTOPPED, 0, .STATUS)
1458          1586  3          ;
1459          1587  3          END
1460          1588  2      ELSE
1461          1589  2      !
1462          1590  2      !  This is a STOP HISTORY command.
1463          1591  2      !
1464          1592  3          BEGIN
1465          1593  3          STR$TRIM (SUBCMDDSC, SUBCMDDSC, SUBCMDDSC [DSC$W_LENGTH] );
1466          1594  3          !
1467          1595  3          ! Dispatch on the subcommand string
1468          1596  3          !
1469          1597  3          SELECTONE TRUE OF
1470          1598  3          SET
1471          1599  3              [ $CMPDSC (SUBCMDDSC, QUAL_HISTORY) ] :
1472          1600  4              (
1473          1601  5              $CMKRNL (ROUTIN = LCPK_STOPHISTORY)
1474          1602  3              );
1475          1603  3
1476          1604  3          [OTHERWISE] : SIGNAL (LCP$_IVCMD);
1477          1605  3
1478          1606  3          TES;
1479          1607  3
1480          1608  3          END
1481          1609  2      ;
1482          1610  2
1483          1611  1      END;
```

```
                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                54  54  00D00 P.AFE:   .ASCII  \TT\                                  ;
                                        00D02          .BLKB   2
                            00000002    00D04 P.AFD:   .LONG   2                                     ;
                            00000000'   00D08          .ADDRESS P.AFE                               ;
                        41  54  4C  5F  00D0C P.AFF:   .ASCII  \_LTA\                               ;

                                                TTNAME=            P.AFD

                                                        .PSECT  $CODE$,NOWRT,2

                                    01FC 00000         .ENTRY  LCP_STOP, Save R2,R3,R4,R5,R6,R7,R8  ; 1499
                    58 00000000G  00  9E 00002         MOVAB   LIB$SIGNAL, R8                       ;
                    57     0000'  CF  9E 00009         MOVAB   QUAL_HISTORY, R7                     ;
                    56 00000000G  00  9E 0000E         MOVAB   SYS$CMKRNL, R6                       ;
                    55     0000'  CF  9E 00015         MOVAB   SUBCMDDSC, R5                        ;
                    5E            28  C2 0001A         SUBL2   #40, SP                              ;
                    65            10  D0 0001D         MOVL    #16, SUBCMDDSC                       ; 1548
                04  A5        F0  A5  9E 00020         MOVAB   SUBCMD, SUBCMDDSC+4                  ;
                                  55  DD 00025         PUSHL   R5                                   ; 1549
                          FF20  C7  9F 00027         PUSHAB  QUAL_SUBCOMMAND                      ;
            00000000G  00       02  FB 0002B         CALLS   #2, CLI$GET_VALUE                    ;
                       52       50  D0 00032         MOVL    R0, STATUS                           ;
            00000000G  8F       52  D1 00035         CMPL    STATUS, #CLI$_ABSENT                 ; 1550
                                73  12 0003C         BNEQ    4$                                   ;
                20  AE          20  D0 0003E         MOVL    #32, DSC                             ; 1559
                24  AE          6E  9E 00042         MOVAB   STR, DSC+4                           ;
                                7E  7C 00046         CLRQ    -(SP)                                ; 1565
                                7E  D4 00048         CLRL    -(SP)                                ;
                            2C  AE  9F 0004A         PUSHAB  DSC                                  ;
                            30  AE  9F 0004D         PUSHAB  DSC                                  ;
                          0C08  C7  9F 00050         PUSHAB  TTNAME                               ;
            00000000G  00       06  FB 00054         CALLS   #6, SYS$TRNLOG                       ;
                       6E   0C10  C7  D1 0005B         CMPL    P.AFF, STR                           ; 1566
                                0C  13 00060         BEQL    1$                                   ;
                                7E  D4 00062         CLRL    -(SP)                                ; 1568
                         0000V  CF  9F 00064         PUSHAB  LCPK_IS_LAT_TERMINAL                 ;
                       66       02  FB 00068         CALLS   #2, SYS$CMKRNL                       ;
                       08       50  E9 0006B         BLBC    R0, 2$                               ;
                          00000000G 8F  DD 0006E 1$: PUSHL   #LCP$_NOTFROMLAT                     ; 1571
                                7E  11 00074         BRB     9$                                   ;
                                7E  D4 00076 2$:     CLRL    -(SP)                                ; 1576
                         0000V  CF  9F 00078         PUSHAB  LCPK_STOPHISTORY                     ;
                       66       02  FB 0007C         CALLS   #2, SYS$CMKRNL                       ;
                                7E  D4 0007F         CLRL    -(SP)                                ; 1581
                         0000V  CF  9F 00081         PUSHAB  LCPK_STOP                            ;
                       66       02  FB 00085         CALLS   #2, SYS$CMKRNL                       ;
                       52       50  D0 00088         MOVL    R0, STATUS                           ;
                       15       52  E9 0008B         BLBC    STATUS, 3$                           ;
                            68  A7  9F 0008E         PUSHAB  QUAL_LOG                             ; 1583
            00000000G  00       01  FB 00091         CALLS   #1, CLI$PRESENT                      ;
                       5C       50  E9 00098         BLBC    R0, 10$                              ;
```

```
                                    00000000G  8F  DD 0009B          PUSHL   #LCP$_STOPPED                                  ;
                                               51  11 000A1          BRB     9$                                             ;
                                               52  DD 000A3 3$:      PUSHL   STATUS                                         ; 1585
                                               7E  D4 000A5          CLRL    -(SP)                                          ;
                                    00000000G  8F  DD 000A7          PUSHL   #LCP$_NOTSTOPPED                               ;
                                 68            03  FB 000AD          CALLS   #3, LIB$SIGNAL                                 ;
                                               04 000B0              RET                                                    ; 1550
                                               55  DD 000B1 4$:      PUSHL   R5                                             ; 1593
                                               55  DD 000B3          PUSHL   R5                                             ;
                                               55  DD 000B5          PUSHL   R5                                             ;
                         00000000G  00         03  FB 000B7          CALLS   #3, STR$TRIM                                   ;
                                    50         65  3C 000BE          MOVZWL  SUBCMDDSC, R0                                  ; 1599
                                    50         67  B1 000C1          CMPW    QUAL_HISTORY, R0                               ;
                                               03  1E 000C4          BGEQU   5$                                             ;
                                    50         67  3C 000C6          MOVZWL  QUAL_HISTORY, R0                               ;
                                    03         50  B1 000C9 5$:      CMPW    R0, #3                                         ;
                                               03  1E 000CC          BGEQU   6$                                             ;
                                    50         03  D0 000CE          MOVL    #3, R0                                         ;
                                               54  D4 000D1 6$:      CLRL    R4                                             ;
          50         20       04  B5         65  2D 000D3          CMPC5   SUBCMDDSC, @SUBCMDDSC+4, #32, R0, -            ;
                                          04  B7     000D9                  @QUAL_HISTORY+4                                ;
                                               02  12 000DB          BNEQ    7$                                             ;
                                               54  D6 000DD          INCL    R4                                             ;
                                    01         54  D1 000DF 7$:      CMPL    R4, #1                                         ;
                                               0A  12 000E2          BNEQ    8$                                             ;
                                               7E  D4 000E4          CLRL    -(SP)                                          ; 1601
                                          0000V  CF  9F 000E6          PUSHAB  LCPK_STOPHISTORY                               ;
                                    66         02  FB 000EA          CALLS   #2, SYS$CMKRNL                                 ;
                                               04 000EC              RET                                                    ; 1600
                                    00000000G  8F  DD 000EE 8$:      PUSHL   #LCP$_IVCMD                                    ; 1604
                                    68         01  FB 000F4 9$:      CALLS   #1, LIB$SIGNAL                                 ;
                                               04 000F7 10$:         RET                                                    ; 1611

; Routine Size:  248 bytes,    Routine Base:  $CODE$ + 05D9
```

```
%SBTTL  'LCP_SET  Set multicast information'
GLOBAL ROUTINE LCP_SET =

!++
! FUNCTIONAL DESCRIPTION:
!
!   The qualifier information is reported and then the kernel mode
!   routine is called to set the multicast information for transmission.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       results of qualifier parse
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       multicast information set
!
!--

    BEGIN

    LITERAL
        STRSIZE = 512,                  ! Size of input and output strings
        PRMSIZE = 24                    ! Size of parameter list in bytes
        ;

    LOCAL
        STATUS,
        DSC : BLOCK [DSC$K_Z_BLN, BYTE],
        STR : VECTOR [STRSIZE,BYTE],
        P : VECTOR [PRMSIZE]
        ;

    !
    !  Get the command qualifiers
    !
    IF NOT (STATUS = LCP_QUAL () )
    THEN
        RETURN .STATUS
    ;

    !
    !  Convert the group codes to an ascii string
    !
```

```
; 1542          1669  2     LCP_CVTGROUPS ();
; 1543          1670  2
; 1544          1671  2     !
; 1545          1672  2     ! Set up the parameter list and report the qualifier information
; 1546          1673  2     ! that we are about to set.
; 1547          1674  2     !
; 1548          1675  2     P[0] = NAMEDSC;
; 1549          1676  2     P[1] = SERVDSC;
; 1550          1677  2     P[2] = NODE_IDDSC;
; 1551          1678  2     P[3] = IDDSC;
; 1552          1679  2     P[4] = GRPDSC;
; 1553          1680  2     P[5] = .MCASTIMR;
; 1554          1681  2
; 1555          1682  2     $SETDSC (DSC, STRSIZE, STR);
; 1556          1683  2
; 1557        P 1684  2     $FAOL
; 1558        P 1685  2         (
; 1559        L 1686  2         CTRSTR = %ASCID
; 1560        L 1687  2         %STRING
; 1561        L 1688  2             (
; 1562        L 1689  2             '!/LCP Qualifier Summary!/!/',
; 1563        L 1690  2             'Node name = !18<\!AS\!>   Service name = !18<\!AS\!>!/',
; 1564        L 1691  2             'Node Identification = \!AS\!/',
; 1565        L 1692  2             'Service Identification = \!AS\!/',
; 1566        L 1693  2             'Groups = !AS!/',
; 1567        L 1694  2             'Multicast timer = !UB seconds!/'
; 1568        P 1695  2             ),
; 1569        P 1696  2         OUTBUF = DSC,
; 1570        P 1697  2         OUTLEN = DSC,
; 1571        P 1698  2         PRMLST = P[0]
; 1572          1699  2         );
; 1573          1700  2
; 1574          1701  2     LOG_OUTPUT (DSC);
; 1575          1702  2
; 1576          1703  2     !
; 1577          1704  2     ! Call the kernel mode routine to set the information in the
; 1578          1705  2     ! driver.
; 1579          1706  2     !
; 1580          1707  3     IF NOT (STATUS = $CMKRNL (ROUTIN = LCPK_SET) )
; 1581          1708  2     THEN
; 1582          1709  2         SIGNAL (LCP$_NOTSET, 0, .STATUS)
; 1583          1710  2     ;
; 1584          1711  2
; 1585          1712  2     !
; 1586          1713  2     !  LOG success if requested
; 1587          1714  2     !
; 1588          1715  2     LOG_SIGNAL (LCP$_SET);
; 1589          1716  2
; 1590          1717  2     SUCCESS
; 1591          1718  2
; 1592          1719  1     END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2
72  65  69  66  69  6C  61  75  51  20  50  43  4C  2F  21  00D10 P.AFH:  .ASCII  \!/LCP Qualifier Summary!/!/Node name = !\  ;
```

```
64 6F 4E 2F 21 2F 21 79 72 61 6D 6D 75 53 20  00D1F
               21 20 3D 20 65 6D 61 6E 20 65  00D2E
65 53 20 20 20 3E 21 5C 53 41 21 5C 3C 38 31  00D38        .ASCII  \18<\<92>\!AS\<92>\!>   Service name = \
      20 3D 20 65 6D 61 6E 20 65 63 69 76 72  00D47
6F 4E 2F 21 3E 21 5C 53 41 21 5C 3C 38 31 21  00D54        .ASCII  \!18<\<92>\!AS\<92>\!>!/Node Identifica\
      61 63 69 66 69 74 6E 65 64 49 20 65 64  00D63
53 2F 21 5C 53 41 21 5C 20 3D 20 6E 6F 69 74  00D70        .ASCII  \tion = \<92>\!AS\<92>\!/Service Identi\
      69 74 6E 65 64 49 20 65 63 69 76 72 65  00D7F
53 41 21 5C 20 3D 20 6E 6F 69 74 61 63 69 66  00D8C        .ASCII  \fication = \<92>\!AS\<92>\!/Groups = !\
      21 20 3D 20 73 70 75 6F 72 47 2F 21 5C  00D9B
74 20 74 73 61 63 69 74 6C 75 4D 2F 21 53 41  00DA8        .ASCII  \AS!/Multicast timer = !UB seconds!/\<0>
6F 63 65 73 20 42 55 21 20 3D 20 72 65 6D 69  00DB7
                              00 2F 21 73 64 6E  00DC6
                                     010E00BB  00DCC P.AFG:  .LONG   17694907
                                     00000000'  00DD0        .ADDRESS P.AFH

                                                            .EXTRN  SYS$FAOL

                                                            .PSECT  $CODE$,NOWRT,2

                                        001C 00000          .ENTRY  LCP_SET, Save R2,R3,R4                  ; 1613
                 54 00000000G  00   9E 00002          MOVAB   LIB$SIGNAL, R4                          ;
                 53 00000000G  00   9E 00009          MOVAB   CLI$PRESENT, R3                         ;
                 5E      FD98  CE   9E 00010          MOVAB   -616(SP), SP                            ;
            FA05 CF            00   FB 00015          CALLS   #0, LCP_QUAL                            ; 1661
                 52            50   D0 0001A          MOVL    R0, STATUS                              ;
                 04            52   E8 0001D          BLBS    STATUS, 1$                              ;
                 50            52   D0 00020          MOVL    STATUS, R0                              ; 1663
                                    04 00023          RET                                             ;
           0000V CF            00   FB 00024 1$:      CALLS   #0, LCP_CVTGROUPS                       ; 1669
                 6E      0000' CF   9E 00029          MOVAB   NAMEDSC, P                              ; 1675
              04 AE      0000' CF   9E 0002E          MOVAB   SERVDSC, P+4                            ; 1676
              08 AE      0000' CF   9E 00034          MOVAB   NODE_IDDSC, P+8                         ; 1677
              0C AE      0000' CF   9E 0003A          MOVAB   IDDSC, P+12                             ; 1678
              10 AE      0000' CF   9E 00040          MOVAB   GRPDSC, P+16                            ; 1679
              14 AE      0000' CF   D0 00046          MOVL    MCASTIMR, P+20                          ; 1680
              F8 AD        0200 8F   3C 0004C          MOVZWL  #512, DSC                               ; 1682
              FC AD          60 AE   9E 00052          MOVAB   STR, DSC+4                              ;
                                 5E  DD 00057          PUSHL   SP                                      ; 1699
                             F8  AD  9F 00059          PUSHAB  DSC                                     ;
                             F8  AD  9F 0005C          PUSHAB  DSC                                     ;
                           0000' CF  9F 0005F          PUSHAB  P.AFG                                   ;
      00000000G  00            04   FB 00063          CALLS   #4, SYS$FAOL                            ;
                           0000' CF  9F 0006A          PUSHAB  QUAL_LOG                                ; 1701
                 63            01   FB 0006E          CALLS   #1, CLI$PRESENT                         ;
                 0A            50   E9 00071          BLBC    R0, 2$                                  ;
                             F8  AD  9F 00074          PUSHAB  DSC                                     ;
      00000000G  00            01   FB 00077          CALLS   #1, LIB$PUT_OUTPUT                      ;
                                 7E  D4 0007E 2$:      CLRL    -(SP)                                   ; 1707
                           0000V CF  9F 00080          PUSHAB  LCPK_SET                                ;
      00000000G  00            02   FB 00084          CALLS   #2, SYS$CMKRNL                          ;
                 52            50   D0 0008B          MOVL    R0, STATUS                              ;
                 0D            52   E8 0008E          BLBS    STATUS, 3$                              ;
                                 52  DD 00091          PUSHL   STATUS                                  ; 1709
                                 7E  D4 00093          CLRL    -(SP)                                   ;
                     00000000G  8F   DD 00095          PUSHL   #LCP$_NOTSET                            ;
                 64            03   FB 0009B          CALLS   #3, LIB$SIGNAL                          ;
```

```
                        0000'  CF  9F 0009E 3$:    PUSHAB  QUAL_LOG                          ; 1715
                 63            01  FB 000A2        CALLS   #1, CLI$PRESENT                   ;
                 09            50  E9 000A5        BLBC    R0, 4$                            ;
                    00000000G  8F  DD 000A8        PUSHL   #LCP$_SET                         ;
                 64            01  FB 000AE        CALLS   #1, LIB$SIGNAL                    ;
                 50            01  D0 000B1 4$:    MOVL    #1, R0                            ; 1719
                                   04 000B4        RET                                       ;

; Routine Size:  181 bytes,    Routine Base:  $CODE$ + 06D1
```

```
1 %SBTTL  'LCP_SHOW  Process a SHOW command'
1 GLOBAL ROUTINE LCP_SHOW : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !   Routine to process an LCP SHOW ... command
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       CLI parsed command
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     LOCAL
2         STATUS
2         ;
2     !
2     ! Obtain the subcommand
2     !
2     $SETDSC (SUBCMDDSC, CMDLEN, SUBCMD);
2     STATUS = CLI$GET_VALUE (QUAL_SUBCOMMAND, SUBCMDDSC);
2     IF .STATUS EQL CLI$_ABSENT
2     THEN
2         SUBCMDDSC [DSC$W_LENGTH] = 0
2     ;
2     STR$TRIM (SUBCMDDSC, SUBCMDDSC, SUBCMDDSC [DSC$W_LENGTH] );
2     !
2     ! Dispatch on the subcommand string
2     !
2     SELECTONE TRUE OF
2     SET
2     [ $CMPDSC (SUBCMDDSC, QUAL_COUNTERS) ] :
3         (
3         LCP_SHOWCOUQUAL ();
3         LCP_SHOWCOU ()
2         );
2
```

```
; 1651   1777  2     [ $CMPDSC (SUBCMDDSC, QUAL_HISTORY) ] :
; 1652   1778  3         (
; 1653   1779  3         LCP_SHOWHIST ()
; 1654   1780  2         );
; 1655   1781  2
; 1656   1782  2     [ $CMPDSC (SUBCMDDSC, QUAL_CHAR) ] :
; 1657   1783  3         (
; 1658   1784  3         LCP_SHOWCHAR ()
; 1659   1785  2         );
; 1660   1786  2
; 1661   1787  2     [ $CMPDSC (SUBCMDDSC, QUAL_SERVERS) ] :
; 1662   1788  3         (
; 1663   1789  3         LCP_SHOWSERVERS ()
; 1664   1790  2         );
; 1665   1791  2
; 1666   1792  2     [OTHERWISE] : SIGNAL (LCP$_IVCMD);
; 1667   1793  2
; 1668   1794  2     TES
; 1669   1795  2
; 1670   1796  1     END;
```

```
                                    01FC 00000          .ENTRY  LCP_SHOW, Save R2,R3,R4,R5,R6,R7,R8          ; 1721
                        58    0000'  CF   9E 00002      MOVAB   SUBCMDDSC, R8                                ;
                        57    0000'  CF   9E 00007      MOVAB   QUAL_COUNTERS, R7                            ;
                        68           10   D0 0000C      MOVL    #16, SUBCMDDSC                               ; 1759
                  04    A8      F0   A8   9E 0000F      MOVAB   SUBCMD, SUBCMDDSC+4                          ;
                                     58   DD 00014      PUSHL   R8                                           ; 1760
                              FF30   C7   9F 00016      PUSHAB  QUAL_SUBCOMMAND                              ;
            00000000G   00           02   FB 0001A      CALLS   #2, CLI$GET_VALUE                            ;
            00000000G   8F           50   D1 00021      CMPL    STATUS, #CLI$_ABSENT                         ; 1761
                                     02   12 00028      BNEQ    1$                                           ;
                                     68   B4 0002A      CLRW    SUBCMDDSC                                    ; 1763
                                     58   DD 0002C 1$:  PUSHL   R8                                           ; 1765
                                     58   DD 0002E      PUSHL   R6                                           ;
                                     58   DD 00030      PUSHL   R8                                           ;
            00000000G   00           03   FB 00032      CALLS   #3, STR$TRIM                                 ;
                        55           68   3C 00039      MOVZWL  SUBCMDDSC, R5                                ; 1771
                        56      04   A8   D0 0003C      MOVL    SUBCMDDSC+4, R6                              ;
                        50           55   D0 00040      MOVL    R5, R0                                       ;
                        50           67   B1 00043      CMPW    QUAL_COUNTERS, R0                            ;
                                     03   1E 00046      BGEQU   2$                                           ;
                        50           67   3C 00048      MOVZWL  QUAL_COUNTERS, R0                            ;
                        03           50   B1 0004B 2$:  CMPW    R0, #3                                       ;
                                     03   1E 0004E      BGEQU   3$                                           ;
                        50           03   D0 00050      MOVL    #3, R0                                       ;
                                     54   D4 00053 3$:  CLRL    R4                                           ;
50        20            66           55   2D 00055      CMPC5   R5, (R6), #32, R0, @QUAL_COUNTERS+4          ;
                                  04 B7      0005A                                                           ;
                                     02   12 0005C      BNEQ    4$                                           ;
                                     54   D6 0005E      INCL    R4                                           ;
                        01           54   D1 00060 4$:  CMPL    R4, #1                                       ;
                                     0B   12 00063      BNEQ    5$                                           ;
                  0000V CF           00   FB 00065      CALLS   #0, LCP_SHOWCOUQUAL                          ; 1773
```

```
                        0000V  CF          00  FB 0006A          CALLS   #0, LCP_SHOWCOU                          ; 1774
                                              04 0006F          RET
                               50          55  D0 00070 5$:     MOVL    R5, R0                                  ; 1777
                               50     10   A7  B1 00073          CMPW    QUAL_HISTORY, R0
                                           04  1E 00077          BGEQU   6$
                               50     10   A7  3C 00079          MOVZWL  QUAL_HISTORY, R0
                               03          50  B1 0007D 6$:     CMPW    R0, #3
                                           03  1E 00080          BGEQU   7$
                               50          03  D0 00082          MOVL    #3, R0
                                           54  D4 00085 7$:     CLRL    R4
        50          20         66          55  2D 00087          CMPC5   R5, (R6), #32, R0, @QUAL_HISTORY+4
                                       14  B7     0008C
                                           02  12 0008E          BNEQ    8$
                                           54  D6 00090          INCL    R4
                               01          54  D1 00092 8$:     CMPL    R4, #1
                                           06  12 00095          BNEQ    9$
                        0000V  CF          00  FB 00097          CALLS   #0, LCP_SHOWHIST                        ; 1779
                                              04 0009C          RET                                             ; 1778
                               50          55  D0 0009D 9$:     MOVL    R5, R0                                  ; 1782
                               50     28   A7  B1 000A0          CMPW    QUAL_CHAR, R0
                                           04  1E 000A4          BGEQU   10$
                               50     28   A7  3C 000A6          MOVZWL  QUAL_CHAR, R0
                               03          50  B1 000AA 10$:     CMPW    R0, #3
                                           03  1E 000AD          BGEQU   11$
                               50          03  D0 000AF          MOVL    #3, R0
                                           54  D4 000B2 11$:     CLRL    R4
        50          20         66          55  2D 000B4          CMPC5   R5, (R6), #32, R0, @QUAL_CHAR+4
                                       2C  B7     000B9
                                           02  12 000BB          BNEQ    12$
                                           54  D6 000BD          INCL    R4
                               01          54  D1 000BF 12$:     CMPL    R4, #1
                                           06  12 000C2          BNEQ    13$
                        0000V  CF          00  FB 000C4          CALLS   #0, LCP_SHOWCHAR                        ; 1784
                                              04 000C9          RET                                             ; 1783
                               50          55  D0 000CA 13$:     MOVL    R5, R0                                  ; 1787
                               50     50   A7  B1 000CD          CMPW    QUAL_SERVERS, R0
                                           04  1E 000D1          BGEQU   14$
                               50     50   A7  3C 000D3          MOVZWL  QUAL_SERVERS, R0
                               03          50  B1 000D7 14$:     CMPW    R0, #3
                                           03  1E 000DA          BGEQU   15$
                               50          03  D0 000DC          MOVL    #3, R0
                                           54  D4 000DF 15$:     CLRL    R4
        50          20         66          55  2D 000E1          CMPC5   R5, (R6), #32, R0, @QUAL_SERVERS+4
                                       54  B7     000E6
                                           02  12 000E8          BNEQ    16$
                                           54  D6 000EA          INCL    R4
                               01          54  D1 000EC 16$:     CMPL    R4, #1
                                           06  12 000EF          BNEQ    17$
                        0000V  CF          00  FB 000F1          CALLS   #0, LCP_SHOWSERVERS                     ; 1789
                                              04 000F6          RET                                             ; 1788
                                  00000000G 8F  DD 000F7 17$:     PUSHL   #LCP$_IVCMD                             ; 1792
                     00000000G  00          01  FB 000FD          CALLS   #1, LIB$SIGNAL
                                              04 00104          RET                                             ; 1796

; Routine Size:  261 bytes,    Routine Base:  $CODE$ + 0786
```

```
; 1672          1797  1 %SBTTL  'LCP_EXIT  Process EXIT LCP request'
; 1673          1798  1 GLOBAL ROUTINE LCP_EXIT : NOVALUE =
; 1674          1799  1
; 1675          1800  1 !++
; 1676          1801  1 ! FUNCTIONAL DESCRIPTION:
; 1677          1802  1 !
; 1678          1803  1 !   Routine to parse an LCP command line.
; 1679          1804  1 !
; 1680          1805  1 ! FORMAL PARAMETERS:
; 1681          1806  1 !
; 1682          1807  1 !       NONE
; 1683          1808  1 !
; 1684          1809  1 ! IMPLICIT INPUTS:
; 1685          1810  1 !
; 1686          1811  1 !       CLI parsed command
; 1687          1812  1 !
; 1688          1813  1 ! IMPLICIT OUTPUTS:
; 1689          1814  1 !
; 1690          1815  1 !       NONE
; 1691          1816  1 !
; 1692          1817  1 ! ROUTINE VALUE:
; 1693          1818  1 ! COMPLETION CODES:
; 1694          1819  1 !
; 1695          1820  1 !       NONE
; 1696          1821  1 !
; 1697          1822  1 ! SIDE EFFECTS:
; 1698          1823  1 !
; 1699          1824  1 !       NONE
; 1700          1825  1 !
; 1701          1826  1 !--
; 1702          1827  1
; 1703          1828  2     BEGIN
; 1704          1829  2
; 1705          1830  2     LOCAL
; 1706          1831  2         STATUS
; 1707          1832  2         ;
; 1708          1833  2
; 1709          1834  3     STATUS = $EXIT (CODE = SS$_NORMAL)
; 1710          1835  3
; 1711          1836  1     END;


                                                        .EXTRN  SYS$EXIT

                                          0000 00000    .ENTRY  LCP_EXIT, Save nothing          ; 1798
                                       01   DD 00002    PUSHL   #1                              ; 1834
                        00000000G  00  01   FB 00004    CALLS   #1, SYS$EXIT                    ;
                                           04 0000B     RET                                     ; 1836

; Routine Size:  12 bytes,    Routine Base:  $CODE$ + 088B
```

```
%SBTTL  'LCPK_SET  Kernel mode routine to set multicast data'
ROUTINE LCPK_SET =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Kernel mode routine to set the information in the buffer of the driver.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       results of the qualifier parse
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       qualifier results set for multicast
!
!--

    BEGIN

    LOCAL
        ZERO,
        ONE,
        TWO,
        THIRTY_TWO,
        LT : REF BLOCK [,BYTE],
        PTR,
        STR : REF BLOCK [,BYTE]
        ;

    ZERO = 0; ONE = 1; TWO = 2; THIRTY_TWO = 32;
    !
    ! Address of the buffer in the driver
    !
    LT = .LCP$L_AREA;
    PTR = STR = .LT [GHB$L_NODE];
    !
    ! Copy the multicast timer, host node status and group codes
    !
    PTR = CH$MOVE (2, MCASTIMR, .PTR);  ! Set timer and no host node status
    PTR = CH$MOVE (1, THIRTY_TWO, .PTR);! Thirty-Two bytes of group code
    PTR = CH$MOVE (32, GROUPS, .PTR);   ! Copy group codes
    !
    ! Copy the host node name and descriptor here
```

```
1894  2      !
1895  2      PTR = CH$MOVE (1, NAMEDSC [DSC$W_LENGTH], .PTR);
1896  2      PTR = CH$MOVE (.NAMEDSC [DSC$W_LENGTH],
1897  2                     .NAMEDSC [DSC$A_POINTER], .PTR);
1898  2      PTR = CH$MOVE (1, NODE_IDDSC [DSC$W_LENGTH], .PTR);
1899  2      PTR = CH$MOVE (.NODE_IDDSC [DSC$W_LENGTH],
1900  2                     .NODE_IDDSC [DSC$A_POINTER], .PTR);
1901  2      !
1902  2      ! If we have an service name, then report 2 service names here
1903  2      !
1904  2      IF .SERVDSC [DSC$W_LENGTH] NEQ 0
1905  2      THEN
1906  2          PTR = CH$MOVE (1, TWO, .PTR)
1907  2      ELSE
1908  2          PTR = CH$MOVE (1, ONE, .PTR)
1909  2      ;
1910  2
1911  2      !
1912  2      ! Copy the first service rating, name and ID here
1913  2      !   We will use the host node name as the first service name
1914  2      !
1915  2      PTR = CH$MOVE (1, RATING, .PTR);
1916  2      PTR = CH$MOVE (1, NAMEDSC [DSC$W_LENGTH], .PTR);
1917  2      PTR = CH$MOVE (.NAMEDSC [DSC$W_LENGTH],
1918  2                     .NAMEDSC [DSC$A_POINTER], .PTR);
1919  2      PTR = CH$MOVE (1, IDDSC [DSC$W_LENGTH], .PTR);
1920  2      PTR = CH$MOVE (.IDDSC [DSC$W_LENGTH], .IDDSC [DSC$A_POINTER], .PTR);
1921  2
1922  2      !
1923  2      ! If there is an service name, then build it and its rating as the
1924  2      !   second service name
1925  2      !
1926  2      IF .SERVDSC [DSC$W_LENGTH] NEQ 0
1927  2      THEN
1928  3          BEGIN
1929  3          PTR = CH$MOVE (1, SERVRATING, .PTR);
1930  3          PTR = CH$MOVE (1, SERVDSC [DSC$W_LENGTH], .PTR);
1931  3          PTR = CH$MOVE (.SERVDSC [DSC$W_LENGTH],
1932  3                         .SERVDSC [DSC$A_POINTER], .PTR);
1933  3          PTR = CH$MOVE (1, ZERO, .PTR)   ! No ID for the second service name!
1934  3          END
1935  2      ;
1936  2
1937  2      !
1938  2      ! Now build the service classes
1939  2      !
1940  2      PTR = CH$MOVE (1, ONE, .PTR);       ! One byte of service class
1941  2      PTR = CH$MOVE (1, ONE, .PTR);       ! Enable service class 0
1942  2
1943  2      !
1944  2      ! Store the total size of the data
1945  2      !
1946  2      STR [-2, 0, 16, 0] = .PTR - .STR;
1947  2
1948  2      !
1949  2      ! Call the set entry of the driver
1950  2      !
```

```
; 1827   1951  2     STARTSTOP (.LT [GHB$L_SETENTRY]);
; 1828   1952  2     !
; 1829   1953  2     ! Ignore the return from the driver for now.  It doesn't mean anything
; 1830   1954  2     !
; 1831   1955  2     SUCCESS
; 1832   1956  2
; 1833   1957  1     END;


                                    0FFC 00000 LCPK_SET:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 1838
                                 7E  D4 00002           CLRL    ZERO                                        ; 1880
                        5B       01  D0 00004           MOVL    #1, ONE                                     ;
                        5A       02  D0 00007           MOVL    #2, TWO                                     ;
                        50       20  D0 0000A           MOVL    #32, THIRTY_TWO                             ;
                        57  0000' CF  D0 0000D          MOVL    LCP$L_AREA, LT                              ; 1884
                        56    1C  A7  D0 00012          MOVL    28(LT), STR                                 ; 1885
                        53       56  D0 00016           MOVL    STR, PTR                                    ;
                        83  0000' CF  B0 00019          MOVW    MCASTIMR, (PTR)+                            ; 1889
                        83       50  90 0001E           MOVB    THIRTY_TWO, (PTR)+                          ; 1890
                63      CF       20  28 00021           MOVC3   #32, GROUPS, (PTR)                          ; 1891
                        83  0000' CF  90 00027          MOVB    NAMEDSC, (PTR)+                             ; 1895
                63      DF  0000' CF  28 0002C          MOVC3   NAMEDSC, @NAMEDSC+4, (PTR)                  ; 1897
                        83  0000' CF  90 00034          MOVB    NODE_IDDSC, (PTR)+                          ; 1898
                63      DF  0000' CF  28 00039          MOVC3   NODE_IDDSC, @NODE_IDDSC+4, (PTR)            ; 1900
                        58  0000' CF  3C 00041          MOVZWL  SERVDSC, R8                                 ; 1904
                                 59  D4 00046           CLRL    R9                                          ;
                                 58  D5 00048           TSTL    R8                                          ;
                                 07  13 0004A           BEQL    1$                                          ;
                                 59  D6 0004C           INCL    R9                                          ;
                        63       5A  90 0004E           MOVB    TWO, (PTR)                                  ; 1906
                                 03  11 00051           BRB     2$                                          ;
                        63       5B  90 00053 1$:       MOVB    ONE, (PTR)                                  ; 1908
                                 53  D6 00056 2$:       INCL    PTR                                         ;
                        83  0000' CF  90 00058          MOVB    RATING, (PTR)+                              ; 1915
                        83  0000' CF  90 0005D          MOVB    NAMEDSC, (PTR)+                             ; 1916
                63      DF  0000' CF  28 00062          MOVC3   NAMEDSC, @NAMEDSC+4, (PTR)                  ; 1918
                        83  0000' CF  90 0006A          MOVB    IDDSC, (PTR)+                               ; 1919
                63      DF  0000' CF  28 0006F          MOVC3   IDDSC, @IDDSC+4, (PTR)                      ; 1920
                        13       59  E9 00077           BLBC    R9, 3$                                      ; 1926
                        83  0000' CF  90 0007A          MOVB    SERVRATING, (PTR)+                          ; 1929
                        83  0000' CF  90 0007F          MOVB    SERVDSC, (PTR)+                             ; 1930
                63      DF       58  28 00084           MOVC3   R8, @SERVDSC+4, (PTR)                       ; 1932
                        83       6E  90 0008A           MOVB    ZERO, (PTR)+                                ; 1933
                        83       5B  90 0008D 3$:       MOVB    ONE, (PTR)+                                 ; 1940
                        83       5B  90 00090           MOVB    ONE, (PTR)+                                 ; 1941
            FE  A6      53       56  A3 00093           SUBW3   STR, PTR, -2(STR)                           ; 1946
                          04     B7  16 00098           JSB     @4(LT)                                      ; 1951
                        50       01  D0 0009B           MOVL    #1, R0                                      ; 1957
                                     04 0009E           RET                                                 ;

; Routine Size:  159 bytes,    Routine Base:  $CODE$ + 0897
```

```
1958  1 %SBTTL  'LCPK_START  Kernel mode start routine'
1959  1 ROUTINE LCPK_START =
1960  1
1961  1 !++
1962  1 ! FUNCTIONAL DESCRIPTION:
1963  1 !
1964  1 !   This routine is called in kernel mode to start the driver.
1965  1 !   First it checks to see if the terminal driver is already active
1966  1 !   by looking at the Timer Active flag.  If
1967  1 !   the device is not active, then a channel is assigned to the
1968  1 !   data link device (a list of possible device names is tried) and
1969  1 !   the ucb for the data link device is passed to the terminal port
1970  1 !   driver.
1971  1 !
1972  1 ! FORMAL PARAMETERS:
1973  1 !
1974  1 !       NONE
1975  1 !
1976  1 ! IMPLICIT INPUTS:
1977  1 !
1978  1 !       LLP$L_AREA
1979  1 !
1980  1 ! IMPLICIT OUTPUTS:
1981  1 !
1982  1 !       NONE
1983  1 !
1984  1 ! ROUTINE VALUE:
1985  1 ! COMPLETION CODES:
1986  1 !
1987  1 !       success or failure code
1988  1 !
1989  1 ! SIDE EFFECTS:
1990  1 !
1991  1 !       NONE
1992  1 !
1993  1 !--
1994  1
1995  2     BEGIN
1996  2
1997  2     LOCAL
1998  2         LT : REF BLOCK [,BYTE],
1999  2         IOSB : VECTOR [4, WORD],
2000  2         CCB : REF BLOCK [,BYTE],
2001  2         UCB : REF BLOCK [,BYTE],
2002  2         ORB : REF BLOCK [,BYTE],
2003  2         DEV,
2004  2         STATUS,
2005  2         DEVCHAN : WORD
2006  2         ;
2007  2
2008  2     !
2009  2     ! If the timer active flag is present, then the port driver is active.
2010  2     !
2011  2     LT = .LCP$L_AREA;
2012  2     IF .LT [GHB$L_TIM_ACT] NEQ 0
2013  2     THEN
2014  2         RETURN SS$_DEVACTIVE
```

```
; 1892          2015  2      ;
; 1893          2016  2      !
; 1894          2017  2      ! Try each of the device names in turn
; 1895          2018  2
; 1896          2019  2      DEV = DEVNAMES;
; 1897          2020  2      WHILE ..DEV NEQ 0
; 1898          2021  2      DO
; 1899          2022  3          BEGIN
; 1900          2023  3          STATUS = $ASSIGN (DEVNAM = ..DEV, CHAN = DEVCHAN);
; 1901          2024  3          IF .STATUS
; 1902          2025  3          THEN
; 1903          2026  3              EXITLOOP
; 1904          2027  3          ;
; 1905          2028  3          DEV = .DEV + 4
; 1906          2029  3          END
; 1907          2030  2      ;
; 1908          2031  2      !
; 1909          2032  2      ! If we never did find a device, then return the error
; 1910          2033  2      !
; 1911          2034  2      IF NOT .STATUS
; 1912          2035  2      THEN
; 1913          2036  2          RETURN .STATUS
; 1914          2037  2      ;
; 1915          2038  2      !
; 1916          2039  2      ! Startup our protocol on the datalink device
; 1917          2040  2      !
; 1918        P 2041  2      STATUS = $QIOW
; 1919        P 2042  2          (
; 1920        P 2043  2          FUNC = IO$_SETMODE OR IO$M_CTRL OR IO$M_STARTUP,
; 1921        P 2044  2          CHAN = .DEVCHAN,
; 1922        P 2045  2          IOSB = IOSB,
; 1923        P 2046  2          P2 = SETPARMDSC
; 1924          2047  2          );
; 1925          2048  2      !
; 1926          2049  2      ! Something bad happened, so deassign our channel and then
; 1927          2050  2      ! return the error
; 1928          2051  2      !
; 1929          2052  2      IF NOT .STATUS
; 1930          2053  2          OR
; 1931          2054  3          NOT ( STATUS = .IOSB [0])
; 1932          2055  2      THEN
; 1933          2056  3          BEGIN
; 1934          2057  3          $DASSGN (CHAN = .DEVCHAN);
; 1935          2058  3          RETURN .STATUS
; 1936          2059  3          END
; 1937          2060  2      ;
; 1938          2061  2      !
; 1939          2062  2      ! Call a handy system routine to find the datalink ucb address
; 1940          2063  2      ! by obtaining the channel control block
; 1941          2064  2      !
; 1942          2065  2      IOC$VERIFYCHAN (.DEVCHAN; CCB);
; 1943          2066  2      UCB = .CCB [CCB$L_UCB];
; 1944          2067  2      !
; 1945          2068  2      ! Call the start entry point of the port driver
; 1946          2069  2      ! and if successful, then fix the ucb so it can stay around
; 1947          2070  2      !
; 1948          2071  2      STATUS = STARTSTOP (.LT [GHB$L_STRENTRY], .UCB);
```

```
; 1949   2072  2     IF .STATUS
; 1950   2073  2     THEN
; 1951   2074  3         BEGIN
; 1952   2075  3         UCB [UCB$L_PID] = 0;
; 1953   2076  3         ORB = .UCB[UCB$L_ORB];
; 1954   2077  3         ORB [ORB$L_OWNER] = %X'10004';
; 1955   2078  3         ORB [ORB$W_PROT] = %X'FFF0'
; 1956   2079  3         END
; 1957   2080  2     ;
; 1958   2081  2     !
; 1959   2082  2     ! Deassign our channel to the datalink ucb
; 1960   2083  2     !
; 1961   2084  2     $DASSGN (CHAN = .DEVCHAN);
; 1962   2085  2     RETURN .STATUS
; 1963   2086  2
; 1964   2087  1     END;


                                                  .EXTRN  SYS$ASSIGN, SYS$QIOW
                                                  .EXTRN  SYS$DASSGN

                                 0FFC 00000 LCPK_START:
                                                  .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 1959
                        5E            10  C2 00002        SUBL2   #16, SP
                        56     0000'  CF  D0 00005        MOVL    LCP$L_AREA, LT                      ; 2011
                                 18   A6  D5 0000A        TSTL    24(LT)                              ; 2012
                                      06  13 0000D        BEQL    1$
                        50     02C4   8F  3C 0000F        MOVZWL  #708, R0                            ; 2014
                                          04 00014        RET
                        52     0000'  CF  9E 00015 1$:    MOVAB   DEVNAMES, DEV                       ; 2019
                                      62  D5 0001A 2$:    TSTL    (DEV)                               ; 2020
                                      19  13 0001C        BEQL    3$
                                      7E  7C 0001E        CLRQ    -(SP)                               ; 2023
                                 0C   AE  9F 00020        PUSHAB  DEVCHAN
                                      62  DD 00023        PUSHL   (DEV)
              00000000G 00           04  FB 00025        CALLS   #4, SYS$ASSIGN
                        57           50  D0 0002C        MOVL    R0, STATUS
                        08           57  E8 0002F        BLBS    STATUS, 4$                          ; 2024
                        52           04  C0 00032        ADDL2   #4, DEV                             ; 2028
                                     E3  11 00035        BRB     2$
                        6A           57  E9 00037 3$:    BLBC    STATUS, 6$                          ; 2034
                                     7E  7C 0003A 4$:    CLRQ    -(SP)                               ; 2047
                                     7E  7C 0003C        CLRQ    -(SP)
                               0000' CF  9F 0003E        PUSHAB  SETPARMDSC
                                     7E  7C 00042        CLRQ    -(SP)
                                     7E  D4 00044        CLRL    -(SP)
                                 28  AE  9F 00046        PUSHAB  IOSB
                        7E     0263  8F  3C 00049        MOVZWL  #611, -(SP)
                        7E       2C  AE  3C 0004E        MOVZWL  DEVCHAN, -(SP)
                                     7E  D4 00052        CLRL    -(SP)
              00000000G 00           0C  FB 00054        CALLS   #12, SYS$QIOW
                        57           50  D0 0005B        MOVL    R0, STATUS
                        38           57  E9 0005E        BLBC    STATUS, 5$                          ; 2052
                        57       08  AE  3C 00061        MOVZWL  IOSB, STATUS                        ; 2054
                        31           57  E9 00065        BLBC    STATUS, 5$
                        50       04  AE  3C 00068        MOVZWL  DEVCHAN, R0                         ; 2065
```

```
                                 00000000G  00  16 0006C          JSB     IOC$VERIFYCHAN                          ;
                              6E            61  D0 00072          MOVL    (CCB), UCB                              ; 2066
                              55            6E  D0 00075          MOVL    UCB, R5                                 ; 2071
                                       10   B6  16 00078          JSB     @16(LT)                                 ;
                              57            50  D0 0007B          MOVL    R0, STATUS                              ;
                              18            57  E9 0007E          BLBC    STATUS, 5$                              ; 2072
                 50           6E            2C  C1 00081          ADDL3   #44, UCB, R0                            ; 2075
                                            60  D4 00085          CLRL    (R0)                                    ;
                 51           6E            1C  C1 00087          ADDL3   #28, UCB, R1                            ; 2076
                              50            61  D0 0008B          MOVL    (R1), ORB                               ;
                              60 00010004   8F  D0 0008E          MOVL    #65540, (ORB)                           ; 2077
                        18    A0            10  AE 00095          MNEGW   #16, 24(ORB)                            ; 2078
                              7E       04   AE  3C 00099 5$:      MOVZWL  DEVCHAN, -(SP)                          ; 2084
                    00000000G 00            01  FB 0009D          CALLS   #1, SYS$DASSGN                          ;
                              50            57  D0 000A4 6$:      MOVL    STATUS, R0                              ; 2085
                                            04  000A7             RET                                             ; 2087

; Routine Size:  168 bytes,    Routine Base:  $CODE$ + 0936
```

```
1 %SBTTL  'LCPK_STOP  Kernel routine to stop the port driver'
1 ROUTINE LCPK_STOP =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !   This routine is called in kernel mode and calls the shut entry point
1 !   in the ltdriver to stop all circuit and run things down.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       LCP$L_AREA
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       success or failure code
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     LITERAL DEVNAMSIZE = 16;
2
2     BIND
2         DEVNAMCTR = DESCRIPTOR ('!AC!UW')
2         ;
2
2     LOCAL
2         LT : REF BLOCK [,BYTE],
2         DEVNAMSTR : VECTOR [DEVNAMSIZE, BYTE],
2         DEVNAMDSC : BLOCK [DSC$K_Z_BLN, BYTE],
2         UCB : REF BLOCK [,BYTE],
2         ORB : REF BLOCK [,BYTE],
2         DDB : REF BLOCK [,BYTE],
2         IOSB : VECTOR [4, WORD],
2         STATUS,
2         DEVCHAN : WORD
2         ;
2
2     !
2     ! If the data link UCB is gone, then we are inactive
2     !
2     LT = .LCP$L_AREA;
2     IF (UCB = .[T [GHB$L_UCB] ) EQL 0
```

```
; 2023          2145  2      THEN
; 2024          2146  2          RETURN SS$_DEVINACT
; 2025          2147  2      ;
; 2026          2148  2      !
; 2027          2149  2      ! Get pointers to the ucb and ddb for datalink to make its name
; 2028          2150  2      ! so we can assign a channel to the data link device ucb
; 2029          2151  2      !
; 2030          2152  2      DDB = .UCB [UCB$L_DDB];
; 2031          2153  2      DEVNAMDSC [DSC$A_POINTER] = DEVNAMSTR;
; 2032          2154  2      DEVNAMDSC [DSC$W_LENGTH] = DEVNAMSIZE;
; 2033        P 2155  2      STATUS = $FAO
; 2034        P 2156  2          (
; 2035        P 2157  2          DEVNAMCTR,
; 2036        P 2158  2          DEVNAMDSC [DSC$W_LENGTH],
; 2037        P 2159  2          DEVNAMDSC,
; 2038        P 2160  2          DDB [DDB$T_NAME],
; 2039        P 2161  2          .UCB [UCB$W_UNIT]
; 2040          2162  2          );
; 2041          2163  2      IF NOT .STATUS
; 2042          2164  2      THEN
; 2043          2165  2          RETURN .STATUS
; 2044          2166  2      ;
; 2045          2167  2      !
; 2046          2168  2      ! Fix the ucb so we can assign a channel to it.
; 2047          2169  2      !
; 2048          2170  2      UCB [UCB$L_PID] = 0;
; 2049          2171  2      ORB = .UCB[UCB$L_ORB];
; 2050          2172  2      ORB [ORB$L_OWNER] = 0;
; 2051          2173  2      ORB [ORB$W_PROT] = 0;
; 2052          2174  2      !
; 2053          2175  2      ! Assign a channel to hold it here and allow us to shut it down
; 2054          2176  2      ! later and deassign the ucb to cause it to go away.
; 2055          2177  2      !
; 2056          2178  2      STATUS = $ASSIGN (DEVNAM = DEVNAMDSC, CHAN = DEVCHAN);
; 2057          2179  2      IF NOT .STATUS
; 2058          2180  2      THEN
; 2059          2181  2          RETURN .STATUS
; 2060          2182  2      ;
; 2061          2183  2      !
; 2062          2184  2      ! Call the shutdown entry of the terminal port driver
; 2063          2185  2      !
; 2064          2186  2      STARTSTOP (.LT [GHB$L_SHUTENTRY]);
; 2065          2187  2      !
; 2066          2188  2      ! Shut down the datalink unit to free the protocol type
; 2067          2189  2      !
; 2068        P 2190  2      STATUS = $QIOW
; 2069        P 2191  2          (
; 2070        P 2192  2          FUNC = IO$_SETMODE OR IO$M_CTRL OR IO$M_SHUTDOWN,
; 2071        P 2193  2          CHAN = .DEVCHAN,
; 2072        P 2194  2          IOSB = IOSB
; 2073          2195  2          );
; 2074          2196  2      !
; 2075          2197  2      ! Collect any failures in status
; 2076          2198  2      !
; 2077          2199  3      IF NOT .STATUS OR NOT (STATUS = .IOSB [0])
; 2078          2200  2      THEN
; 2079          2201  2          0
```

```
; 2080   2202  2     ;
; 2081   2203  2
; 2082   2204  2     !
; 2083   2205  2     ! Temporary code to clean up io queue for mailbox delete
; 2084   2206  2     !
; 2085   2207  2     ! UCB [UCB$L_IOBL] = UCB [UCB$L_IOFL] = 0;
; 2086   2208  2
; 2087   2209  2     !
; 2088   2210  2     ! Deassign the channel that we have.  If shutentry succeeded, then
; 2089   2211  2     ! the device will disappear
; 2090   2212  2     !
; 2091   2213  2     $DASSGN (CHAN = .DEVCHAN);
; 2092   2214  2
; 2093   2215  2     RETURN .STATUS
; 2094   2216  2
; 2095   2217  1     END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                57  55  21  43  41  21  00DD4 P.AFJ:   .ASCII  \!AC!UW\                                  ;
                                        00DDA          .BLKB   2
                            00000006    00DDC P.AFI:   .LONG   6                                         ;
                            00000000'   00DE0          .ADDRESS P.AFJ                                   ;

                                                DEVNAMCTR=          P.AFI
                                                        .EXTRN  SYS$FAO

                                                        .PSECT  $CODE$,NOWRT,2

                                        0FFC 00000 LCPK_STOP:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11     ; 2089
            5E                  28  C2 00002           SUBL2   #40, SP                                  ;
            53          0000'   CF  D0 00005           MOVL    LCP$L_AREA, LT                           ; 2143
            52            14    A3  D0 0000A           MOVL    20(LT), UCB                              ; 2144
                                06  12 0000E           BNEQ    1$                                       ;
            50          20D4    8F  3C 00010           MOVZWL  #8404, R0                                ; 2146
                                    04 00015           RET                                              ;
            50            28    A2  D0 00016 1$:       MOVL    40(UCB), DDB                             ; 2152
        14  AE            18    AE  9E 0001A           MOVAB   DEVNAMSTR, DEVNAMDSC+4                   ; 2153
        10  AE                  10  B0 0001F           MOVW    #16, DEVNAMDSC                           ; 2154
            7E            54    A2  3C 00023           MOVZWL  84(UCB), -(SP)                           ; 2162
                          14    A0  9F 00027           PUSHAB  20(DDB)                                  ;
                          18    AE  9F 0002A           PUSHAB  DEVNAMDSC                                ;
                          1C    AE  9F 0002D           PUSHAB  DEVNAMDSC                                ;
                        0000'   CF  9F 00030           PUSHAB  DEVNAMCTR                                ;
    00000000G   00              05  FB 00034           CALLS   #5, SYS$FAO                              ;
            6E                  50  D0 0003B           MOVL    R0, STATUS                               ;
            56                  6E  E9 0003E           BLBC    STATUS, 3$                               ; 2163
                          2C    A2  D4 00041           CLRL    44(UCB)                                  ; 2170
            50            1C    A2  D0 00044           MOVL    28(UCB), ORB                             ; 2171
                                60  D4 00048           CLRL    (ORB)                                    ; 2172
                          18    A0  B4 0004A           CLRW    24(ORB)                                  ; 2173
                                7E  7C 0004D           CLRQ    -(SP)                                    ; 2178
                          0C    AE  9F 0004F           PUSHAB  DEVCHAN                                  ;
                          1C    AE  9F 00052           PUSHAB  DEVNAMDSC                                ;
```

```
                    00000000G  00          04  FB 00055          CALLS   #4, SYS$ASSIGN
                               6E          50  D0 0005C          MOVL    R0, STATUS
                               35          6E  E9 0005F          BLBC    STATUS, 3$                          ; 2179
                                       0C  B3  16 00062          JSB     @12(LT)                             ; 2186
                                           7E  7C 00065          CLRQ    -(SP)                               ; 2195
                                           7E  7C 00067          CLRQ    -(SP)
                                           7E  7C 00069          CLRQ    -(SP)
                                           7E  7C 0006B          CLRQ    -(SP)
                                       28  AE  9F 0006D          PUSHAB  IOSB
                               7E    02A3  8F  3C 00070          MOVZWL  #675, -(SP)
                               7E      2C  AE  3C 00075          MOVZWL  DEVCHAN, -(SP)
                                           7E  D4 00079          CLRL    -(SP)
                    00000000G  00          0C  FB 0007B          CALLS   #12, SYS$QIOW
                               6E          50  D0 00082          MOVL    R0, STATUS
                               04          6E  E9 00085          BLBC    STATUS, 2$                          ; 2199
                               6E      08  AE  3C 00088          MOVZWL  IOSB, STATUS
                               7E      04  AE  3C 0008C 2$:      MOVZWL  DEVCHAN, -(SP)                      ; 2213
                    0000C000G  00          01  FB 00090          CALLS   #1, SYS$DASSGN
                               50          6E  D0 00097 3$:      MOVL    STATUS, R0                          ; 2215
                                               04 0009A          RET                                         ; 2217

; Routine Size:  155 bytes,    Routine Base:  $CODE$ + 09DE
```

```
%SBTTL  'LCPK_STARTHISTORY  Kernel mode start history routine'
ROUTINE LCPK_STARTHISTORY =

!++
! FUNCTIONAL DESCRIPTION:
!
!   This routine is called in kernel mode to start history journalling
!   in the driver. First it checks to see if the terminal driver is
!   already active by looking at the Timer Active flag.  If
!   the device is not active, then an error is returned, else the history
!   buffer is allocated and pointed to in the driver global area.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       LCP$L_AREA
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       success or failure code
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        LT : REF BLOCK [,BYTE],
        HIST_BUF : REF BLOCK [,BYTE],
        STATUS
        ;

    !
    !  Check to make sure the driver is active.
    !
    LT = .LCP$L_AREA;
    IF .LT [GHB$L_TIM_ACT] EQL 0
    THEN
        RETURN SS$_DEVINACT
    ;

    !
    !  Try to allocate a buffer, else insufficient memory
    !
    IF NOT (STATUS = EXE$ALONONPAGED (GHB$K_HISTSIZE+HBF_C_LENGTH; HIST_BUF) )
    THEN
```

```
; 2154   2275  2          RETURN SS$_INSFMEM
; 2155   2276  2      ;
; 2156   2277  2
; 2157   2278  2      !
; 2158   2279  2      !  Initialize the buffer and move pointer into global area
; 2159   2280  2      !
; 2160   2281  2      HIST_BUF[HBF_L_NFXT] = HIST_BUF[HBF_Z_DATA];
; 2161   2282  2      HIST_BUF[HBF_L_BUFEND] = HIST_BUF[HBF_Z_DATA] + GHB$K_HISTSIZE;
; 2162   2283  2      HIST_BUF[HBF_W_SIZE] = GHB$K_HISTSIZE;
; 2163   2284  2      HIST_BUF[HBF_B_TYPE] = DYN$C_BUFIO;
; 2164   2285  2      CH$FILL (0, GHB$K_HISTSIZE, HIST_BUF [HBF_Z_DATA]); ! Zero the buffer
; 2165   2286  2
; 2166   2287  2      LT [GHB$L_HISTORY] = HIST_BUF [HBF_L_NEXT];
; 2167   2288  2
; 2168   2289  2      SUCCESS
; 2169   2290  2
; 2170   2291  1      END;


                                    0FFC 00000 LCPK_STARTHISTORY:
                                                       .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 2219
                        57     0000'  CF  D0 00002     MOVL    LCP$L_AREA, LT                                ; 2264
                                 18   A7  D5 00007     TSTL    24(LT)                                        ; 2265
                                      06  12 0000A     BNEQ    1$                                            ;
                        50     20D4   8F  3C 0000C     MOVZWL  #8404, R0                                     ; 2267
                                          04 00011     RET                                                   ;
                        51     0A0D   8F  3C 00012 1$: MOVZWL  #2573, R1                                     ; 2273
                           00000000G   00  16 00017     JSB     EXE$ALONONPAGED                               ;
                        56             52  D0 0001D     MOVL    R2, R6                                        ;
                        06             50  E8 00020     BLBS    STATUS, 2$                                    ;
                        50     0124   8F  3C 00023     MOVZWL  #292, R0                                      ; 2275
                                          04 00028     RET                                                   ;
                        66       0C   A6  9E 00029 2$: MOVAB   12(HIST_BUF), (HIST_BUF)                      ; 2281
                     04 A6     0A0C   C6  9E 0002D     MOVAB   2572(R6), 4(HIST_BUF)                         ; 2282
                     08 A6     0A00   8F  B0 00033     MOVW    #2560, 8(HIST_BUF)                            ; 2283
                     0A A6            13  90 00039     MOVB    #19, 10(HIST_BUF)                             ; 2284
   0A00  8F      00       6E          00  2C 0003D     MOVC5   #0, (SP), #0, #2560, 12(HIST_BUF)             ; 2285
                                 0C   A6     00044                                                           ;
                     08 A7            56  D0 00046     MOVL    HIST_BUF, 8(LT)                               ; 2287
                        50            01  D0 0004A     MOVL    #1, R0                                        ; 2291
                                          04 0004D     RET                                                   ;

; Routine Size:  78 bytes,    Routine Base:  $CODE$ + 0A79
```

```
%SBTTL  'LCPK_STOPHISTORY  Kernel mode stop history routine'
ROUTINE LCPK_STOPHISTORY : NOVALUE  =

!++
! FUNCTIONAL DESCRIPTION:
!
!   This routine is called in kernel mode to start history journalling
!   in the driver. First it checks to see if the terminal driver is
!   already active by looking at the Timer Active flag.  If
!   the device is not active, then an error is returned, else the history
!   buffer is allocated and pointed to in the driver global area.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       LCP$L_AREA
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        HIST_BUF : REF BLOCK [,BYTE]
        ;

    !
    !  Check to make sure the driver is active.
    !
    IF .LCP$L_AREA [GHB$L_HISTORY] EQL 0
    THEN
        RETURN
    ;

    !
    !  Get history buffer and deallocate it.
    !
    HIST_BUF = .LCP$L_AREA [GHB$L_HISTORY];
    LCP$L_AREA [GHB$L_HISTORY] = 0;
    EXE$DEANONPAGED (HIST_BUF);

    END;
```

```
                                  0FFC 00000 LCPK_STOPHISTORY:
                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 2293
                    5E              04  C2 00002        SUBL2   #4, SP                              ;
                    50        0000' CF  D0 00005        MOVL    LCP$L_AREA, R0                      ; 2336
                                 08 A0  D5 0000A        TSTL    8(R0)                               ;
                                    10  13 0000D        BEQL    1$                                  ;
                    6E           08 A0  D0 0000F        MOVL    8(R0), HIST_BUF                     ; 2344
                                 08 A0  D4 00013        CLRL    8(R0)                               ; 2345
                    50              6E  9E 00016        MOVAB   HIST_BUF, R0                        ; 2346
                       00000000G    00  16 00019        JSB     EXE$DEANONPAGED                     ;
                                        04 0001F 1$:    RET                                         ; 2348

; Routine Size:  32 bytes,    Routine Base:  $CODE$ + 0AC7
```

```
%SBTTL  'LCPK_IS_LAT_TERMINAL  Kernel routine to determine if TT is LAT terminal'
ROUTINE LCPK_IS_LAT_TERMINAL =

!++
! FUNCTIONAL DESCRIPTION:
!
!   This routine is called in kernel mode to determine if the current
!   TT: device is a LAT terminal.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       success or failure code
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    BIND
        TTNAME  = DESCRIPTOR ('TT'),
        DRVRNAM = DESCRIPTOR ('LTDRIVER') : BLOCK [,BYTE]
        ;

    LOCAL
        UCB : REF BLOCK [,BYTE],
        CCB : REF BLOCK [,BYTE],
        DDB : REF BLOCK [,BYTE],
        STATUS,
        DEVCHAN : WORD
        ;

    !
    !  If we can't assign a channel, then return success, to prevent
    !  shutting down LAT.
    !
    IF NOT (STATUS = $ASSIGN (DEVNAM = TTNAME, CHAN = DEVCHAN) )
    THEN
        RETURN SUCCESS
    ;

    !
```

```
; 2287   2406  2      !  Call a handy system routine to find the terminal ucb address
; 2288   2407  2      !  from the channel control block.
; 2289   2408  2      !
; 2290   2409  2      IOC$VERIFYCHAN (.DEVCHAN; CCB);
; 2291   2410  2      UCB = .CCB [CCB$L_UCB];                    ! Get (virtual?) UCB address
; 2292   2411  2      !
; 2293   2412  2      ! Deassign our channel to the terminal ucb
; 2294   2413  2      !
; 2295   2414  2      $DASSGN (CHAN = .DEVCHAN);
; 2296   2415  2
; 2297   2416  2      IF .UCB [UCB$L_TL_PHYUCB] EQL 0            ! LT devices should have this
; 2298   2417  2      THEN                                       !    field set!
; 2299   2418  2          RETURN FAILURE                         ! Assume we're not an LT term
; 2300   2419  2      ;
; 2301   2420  2
; 2302   2421  2      UCB = .UCB [UCB$L_TL_PHYUCB];              ! Get Physical UCB address
; 2303   2422  2      DDB = .UCB [UCB$L_DDB];                    ! Get DDB address
; 2304   2423  2
; 2305   2424  2      !
; 2306   2425  2      !  Return the result of the comparison
; 2307   2426  2      !
; 2308   2427  2      CH$EQL
; 2309   2428  2          (
; 2310   2429  2          .DRVRNAM [DSC$W_LENGTH], .DRVRNAM [DSC$A_POINTER],
; 2311   2430  2          CH$RCHAR (DDB [DDB$B_DRVNAM_LEN]),
; 2312   2431  2          DDB [DDB$T_DRVNAME] +1,
; 2313   2432  2          0
; 2314   2433  2          )
; 2315   2434  2
; 2316   2435  1      END;


                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                                 54  54  00DE4 P.AFL:  .ASCII  \TT\                                  ;
                                         00DE6         .BLKB   2
                             00000002    00DE8 P.AFK:  .LONG   2                                     ;
                             00000000'   00DEC         .ADDRESS P.AFL                              ;
52  45  56  49  52  44  54  4C          00DF0 P.AFN:  .ASCII  \LTDRIVER\                           ;
                             00000008    00DF8 P.AFM:  .LONG   8                                     ;
                             00000000'   00DFC         .ADDRESS P.AFN                              ;

                                                 TTNAME=            P.AFK
                                                 DRVRNAM=           P.AFM


                                                    .PSECT  $CODE$,NOWRT,2

                                   0FFC 00000 LCPK_IS_LAT_TERMINAL:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 2350
            5E                 04  C2 00002         SUBL2   #4, SP                                      ;
                               7E  7C 00005         CLRQ    -(SP)                                       ; 2400
                     08        AE  9F 00007         PUSHAB  DEVCHAN                                     ;
                   0000'       CF  9F 0000A         PUSHAB  TTNAME                                      ;
00000000G   00                 04  FB 0000E         CALLS   #4, SYS$ASSIGN                              ;
            04                 50  E8 00015         BLBS    STATUS, 1$                                  ;
```

```
                                50          01  D0 00018          MOVL    #1, R0                                  ; 2402
                                                04 0001B          RET                                             ;
                                50          6E  3C 0001C 1$:      MOVZWL  DEVCHAN, R0                             ; 2409
                          00000000G         00  16 0001F          JSB     IOC$VERIFYCHAN                          ;
                                52          61  D0 00025          MOVL    (CCB), UCB                              ; 2410
                                7E          6E  3C 00028          MOVZWL  DEVCHAN, -(SP)                          ; 2414
                   00000000G    00          01  FB 0002B          CALLS   #1, SYS$DASSGN                          ;
                                50     00A0 C2  D0 00032          MOVL    160(UCB), R0                            ; 2416
                                            03  12 00037          BNEQ    2$                                      ;
                                            50  D4 00039          CLRL    R0                                      ; 2418
                                                04 0003B          RET                                             ;
                                52          50  D0 0003C 2$:      MOVL    R0, UCB                                 ; 2421
                                50       28 A2  D0 0003F          MOVL    40(UCB), DDB                            ; 2422
                                51       24 A0  9A 00043          MOVZBL  36(DDB), R1                             ; 2430
                                            54  D4 00047          CLRL    R4                                      ;
     51         00     0000'   DF     0000' CF  2D 00049          CMPC5   DRVRNAM, @DRVRNAM+4, #0, R1, 37(DDB)    ;
                                       25   A0     00052                                                          ;
                                            02  12 00054          BNEQ    3$                                      ;
                                            54  D6 00056          INCL    R4                                      ;
                                50          54  D0 00058 3$:      MOVL    R4, R0                                  ; 2435
                                                04 0005B          RET                                             ;

; Routine Size:  92 bytes,    Routine Base:  $CODE$ + 0AE7
```

```
; 2318          2436  1 %SBTTL  'LCP_ZERO  Zero counters and history buffer'
; 2319          2437  1 GLOBAL ROUTINE LCP_ZERO : NOVALUE =
; 2320          2438  1
; 2321          2439  1 !++
; 2322          2440  1 ! FUNCTIONAL DESCRIPTION:
; 2323          2441  1 !
; 2324          2442  1 !   Call the kernel mode routine to zero the counters and history buffer.
; 2325          2443  1 !
; 2326          2444  1 ! FORMAL PARAMETERS:
; 2327          2445  1 !
; 2328          2446  1 !       NONE
; 2329          2447  1 !
; 2330          2448  1 ! IMPLICIT INPUTS:
; 2331          2449  1 !
; 2332          2450  1 !       NONE
; 2333          2451  1 !
; 2334          2452  1 ! IMPLICIT OUTPUTS:
; 2335          2453  1 !
; 2336          2454  1 !       NONE
; 2337          2455  1 !
; 2338          2456  1 ! ROUTINE VALUE:
; 2339          2457  1 ! COMPLETION CODES:
; 2340          2458  1 !
; 2341          2459  1 !       NONE
; 2342          2460  1 !
; 2343          2461  1 ! SIDE EFFECTS:
; 2344          2462  1 !
; 2345          2463  1 !       NONE
; 2346          2464  1 !
; 2347          2465  1 !--
; 2348          2466  1
; 2349          2467  2     BEGIN
; 2350          2468  2
; 2351          2469  3     $CMKRNL (ROUTIN = LCPK_ZERO)
; 2352          2470  3
; 2353          2471  1     END;


                                        0000 00000          .ENTRY  LCP_ZERO, Save nothing          ; 2437
                                     7E   D4 00002          CLRL    -(SP)                           ; 2469
                                0000V CF   9F 00004          PUSHAB  LCPK_ZERO                       ;
                     00000000G  00         02 FB 00008       CALLS   #2, SYS$CMKRNL                  ;
                                              04 0000F       RET                                     ; 2471

; Routine Size:  16 bytes,    Routine Base:  $CODE$ + 0B43
```

```
%SBTTL  'LCPK_ZERO  Kernel mode zero routine'
ROUTINE LCPK_ZERO : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Zero the counter block and the history buffer
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        LT : REF BLOCK [,BYTE],
        HIST_BUF : REF BLOCK [,BYTE]
        ;

    LT = .LCP$L_AREA;
    !
    ! Zero the counterblock
    !
    CH$FILL (0, GHB$K_CTRLENGTH, LT [GHB$T_COUNTERS] );

    !
    ! Zero the history buffer if there is one
    !
    IF .LT [GHB$L_HISTORY] NEQ 0
    THEN
        BEGIN
        HIST_BUF = .LT [GHB$L_HISTORY];
        CH$FILL (0, GHB$K_HISTSIZE, HIST_BUF [HBF_Z_DATA]);
        END
    ;

    END;
```

```
                                                 007C 00000 LCPK_ZERO:
                                                                    .WORD   Save R2,R3,R4,R5,R6                          ; 2473
                                56      0000'  CF  D0 00002         MOVL    LCP$L_AREA, LT                               ; 2510
        34          00          6E             00  2C 00007         MOVC5   #0, (SP), #0, #52, 44(LT)                    ; 2514
                                          2C  A6     0000C                                                               ;
                                          08  A6  D5 0000E         TSTL    8(LT)                                        ; 2519
                                              0D  13 00011         BEQL    1$                                           ;
                                50        08  A6  D0 00013         MOVL    8(LT), HIST_BUF                              ; 2522
    0A00    8F          00      6E             00  2C 00017         MOVC5   #0, (SP), #0, #2560, 12(HIST_BUF)            ; 2523
                                          0C  A0     0001E                                                               ;
                                                 04 00020 1$:       RET                                                  ; 2527

; Routine Size:  33 bytes,    Routine Base:  $CODE$ + 0B53
```

```
%SBTTL  'LCP_SHOWCOUQUAL  Process show qualifiers'
ROUTINE LCP_SHOWCOUQUAL : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Process qualifiers for the SHOW COUNTERS command.  Convert the qualifiers
!   to descriptors to set in the driver.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       cli processed qualifiers
!
! IMPLICIT OUTPUTS:
!
!       qualifier descriptors
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       errors signalled
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    !
    !  All qualifiers are optional!
    !
    FLAGS [FLAG_V_NODE] = CLI$PRESENT (QUAL_NODE);
    FLAGS [FLAG_V_DEVICE] = CLI$PRESENT (QUAL_DEVICE);
    FLAGS [FLAG_V_SERVERS] = CLI$PRESENT (QUAL_SERVERS);

    !
    !  If no qualifier is specified, then default to /NODE
    !
    IF NOT (
            .FLAGS [FLAG_V_NODE] OR
            .FLAGS [FLAG_V_DEVICE] OR
            .FLAGS [FLAG_V_SERVERS]
            )
    THEN
        FLAGS [FLAG_V_NODE] = TRUE
    ;

    END;
```

```
                                        000C 00000 LCP_SHOWCOUQUAL:
                                                        .WORD   Save R2,R3                          ; 2529
                          53 00000000G  00  9E 00002    MOVAB   CLI$PRESENT, R3                     ;
                          52     0000'  CF  9E 00009    MOVAB   FLAGS, R2                           ;
                                0000'  CF  9F 0000E    PUSHAB  QUAL_NODE                           ; 2565
                          63           01  FB 00012    CALLS   #1, CLI$PRESENT                     ;
        62        01      02           50  F0 00015    INSV    R0, #2, #1, FLAGS                   ;
                                0000'  CF  9F 0001A    PUSHAB  QUAL_DEVICE                         ; 2566
                          63           01  FB 0001E    CALLS   #1, CLI$PRESENT                     ;
        62        01      01           50  F0 00021    INSV    R0, #1, #1, FLAGS                   ;
                                0000'  CF  9F 00026    PUSHAB  QUAL_SERVERS                        ; 2567
                          63           01  FB 0002A    CALLS   #1, CLI$PRESENT                     ;
        62        01      00           50  F0 0002D    INSV    R0, #0, #1, FLAGS                   ;
                  0A      62           02  E0 00032    BBS     #2, FLAGS, 1$                       ; 2573
                  06      62           01  E0 00036    BBS     #1, FLAGS, 1$                       ; 2574
                          03           62  E8 0003A    BLBS    FLAGS, 1$                           ; 2575
                          62           04  88 0003D    BISB2   #4, FLAGS                           ; 2578
                                          04 00040 1$:  RET                                         ; 2581

; Routine Size:  65 bytes,    Routine Base:  $CODE$ + 0B74
```

```
%SBTTL  'LCP_SHOWCOU  Process show counters request'
ROUTINE LCP_SHOWCOU : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Convert and write the requested counters to sys$output.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       counters in LAT driver or Ethernet device driver
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       errors signalled
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    !
    !  If the /NODE qualifier is present, then display Host Node counters.
    !
    IF .FLAGS [FLAG_V_NODE]
    THEN
        LCP_SHOWNODECOU ()
    ;

    !
    !  If the /DEVICE qualifier is present, then display Ethernet device
    !  counters.
    !
    IF .FLAGS [FLAG_V_DEVICE]
    THEN
        LCP_SHOWDEVCOU ()
    ;

    !
    !  If the /SERVERS qualifier is present, then display SERVER counters.
    !
    IF .FLAGS [FLAG_V_SERVERS]
    THEN
        LCP_SHOWSERVCOU ()
    ;
```

```
; 2524          2639  2
; 2525          2640  1     END;


                                        0004 00000 LCP_SHOWCOU:
                                                        .WORD   Save R2                                  ; 2583
                        52     0000'  CF  9E 00002      MOVAB   FLAGS, R2                                ;
                 05     62            02  E1 00007      BBC     #2, FLAGS, 1$                            ; 2618
                 0000V  CF            00  FB 0000B      CALLS   #0, LCP_SHOWNODECOU                      ; 2620
                 05     62            01  E1 00010 1$:  BBC     #1, FLAGS, 2$                            ; 2627
                 0000V  CF            00  FB 00014      CALLS   #0, LCP_SHOWDEVCOU                       ; 2629
                        05            62  E9 00019 2$:  BLBC    FLAGS, 3$                                ; 2635
                 0000V  CF            00  FB 0001C      CALLS   #0, LCP_SHOWSERVCOU                      ; 2637
                                         04 00021 3$:  RET                                              ; 2640

; Routine Size:  34 bytes,    Routine Base:  $CODE$ + 0BB5
```

```
%SBTTL  'LCP_SHOWNODECOU  Show Host Node counters'
ROUTINE LCP_SHOWNODECOU : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Convert and write the NODE counters to sys$output.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       counters in LAT driver
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        LT : REF BLOCK [,BYTE],
        PTR : REF VECTOR [],
        STATUS
        ;


    CH$FILL (255, GHB$S_COMM_AREA, AREA);

    !
    ! Get the counters if we can, and if not then signal the error
    !
    IF NOT
        (STATUS = $CMKRNL (ROUTIN = LCPK_GETCOUNTERS) )
    THEN
        BEGIN
        SIGNAL (.STATUS);
        RETURN
        END
    ;

    !
    ! convert the counters for output in a single bound
    !
```

```
; 2584          2698  2     $SETDSC (OUTDSC, OUTBUFSIZE, OUTBUF);
; 2585          2699  2
; 2586        P 2700  2     STATUS = $FAOL
; 2587        P 27    2         (
; 2588        P 27    2         CTRSTR = COUNTERSCTRSTR,
; 2589        P 2703  2         OUTLEN = OUTDSC [DSC$W_LENGTH],
; 2590        P 2704  2         OUTBUF = OUTDSC,
; 2591        P 2705  2         PRMLST = AREA [GHB$T_COUNTERS]
; 2592          2706  2         );
; 2593          2707  2
; 2594          2708  2     IF NOT .STATUS THEN SIGNAL (.STATUS);
; 2595          2709  2
; 2596          2710  2     STATUS = LIB$PUT_OUTPUT (OUTDSC);
; 2597          2711  2
; 2598          2712  2     IF NOT .STATUS THEN SIGNAL (.STATUS);
; 2599          2713  2
; 2600          2714  2     !
; 2601          2715  2     ! Print the mask bit names that are present
; 2602          2716  2     !
; 2603          2717  2     LT = AREA;
; 2604          2718  2     IF .LT [GHB$L_PROTOMASK] NEQ 0
; 2605          2719  2     THEN
; 2606          2720  3         BEGIN
; 2607          2721  3         STATUS = LIB$PUT_OUTPUT (PROTOCOLMASK);
; 2608          2722  3         PTR = PROTOCOLBITS;
; 2609          2723  3         INCR IDX FROM 0 TO 64 BY 2
; 2610          2724  3         DO
; 2611          2725  4             BEGIN
; 2612          2726  4             !
; 2613          2727  4             ! If the string desc is blank, then exit loop
; 2614          2728  4             !
; 2615          2729  4             IF .PTR [.IDX + 1] EQL 0
; 2616          2730  4             THEN
; 2617          2731  4                 EXITLOOP
; 2618          2732  4             ;
; 2619          2733  4             !
; 2620          2734  4             ! If the bit is set, then write the string
; 2621          2735  4             !
; 2622          2736  4             IF .LT [$BYTEOFFSET (GHB$L_PROTOMASK), .PTR [.IDX], 1, 0]
; 2623          2737  4             THEN
; 2624          2738  4                 LIB$PUT_OUTPUT (.PTR [.IDX + 1] )
; 2625          2739  4             END
; 2626          2740  3         ;
; 2627          2741  3         END
; 2628          2742  2     ;
; 2629          2743  2
; 2630          2744  1     END;
```

```
                    01FC 00000 LCP_SHOWNODECOU:
                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8                    ; 2642
   58 00000000G  00  9E 00002           MOVAB   LIB$PUT_OUTPUT, R8                           ;
   57 00000000G  00  9E 00009           MOVAB   LIB$SIGNAL, R7                               ;
   56     0000'  CF  9E 00010           MOVAB   OUTDSC, R6                                   ;
```

```
0060  8F  FF  8F          6E          00  2C 00015          MOVC5   #0, (SP), #255, #96, AREA          ; 2681
                               F5FC  C6     0001D
                                     7E  D4 00020          CLRL    -(SP)                              ; 2687
                               0000V  CF  9F 00022          PUSHAB  LCPK_GETCOUNTERS
                     00000000G  00     02  FB 00026          CALLS   #2, SYS$CMKRNL
                                52     50  D0 0002D          MOVL    R0, STATUS
                                06     52  E8 00030          BLBS    STATUS, 1$
                                       52  DD 00033          PUSHL   STATUS                             ; 2690
                                67     01  FB 00035          CALLS   #1, LIB$SIGNAL
                                       04  00038          RET                                        ; 2689
                                66  0400  8F  3C 00039 1$:   MOVZWL  #1024, OUTDSC                      ; 2698
                             04 A6    08  A6  9E 0003E      MOVAB   OUTBUF, OUTDSC+4
                               F628  C6  9F 00043          PUSHAB  AREA+44                            ; 2706
                                     56  DD 00047          PUSHL   R6
                                     56  DD 00049          PUSHL   R6
                               0000' CF  9F 0004B          PUSHAB  COUNTERSCTRSTR
                     00000000G  00     04  FB 0004F          CALLS   #4, SYS$FAOL
                                52     50  D0 00056          MOVL    R0, STATUS
                                05     52  E8 00059          BLBS    STATUS, 2$                         ; 2708
                                       52  DD 0005C          PUSHL   STATUS
                                67     01  FB 0005E          CALLS   #1, LIB$SIGNAL
                                       56  DD 00061 2$:      PUSHL   R6                                 ; 2710
                                68     01  FB 00063          CALLS   #1, LIB$PUT_OUTPUT
                                52     50  D0 00066          MOVL    R0, STATUS
                                05     52  E8 00069          BLBS    STATUS, 3$                         ; 2712
                                       52  DD 0006C          PUSHL   STATUS
                                67     01  FB 0006E          CALLS   #1, LIB$SIGNAL
                                53  F5FC  C6  9E 00071 3$:   MOVAB   AREA, LT                          ; 2717
                                    50  A3  D5 00076          TSTL    80(LT)                             ; 2718
                                       2D  13 00079          BEQL    6$
                               0000' CF  9F 0007B          PUSHAB  PROTOCOLMASK                       ; 2721
                                68     01  FB 0007F          CALLS   #1, LIB$PUT_OUTPUT
                                52     50  D0 00082          MOVL    R0, STATUS
                                54  0000' CF  9E 00085      MOVAB   PROTOCOLBITS, PTR                  ; 2722
                                       52  D4 0008A          CLRL    IDX                                ; 2729
                                50  04 A442  D0 0008C 4$:   MOVL    4(PTR)[IDX], R0
                                       15  13 00091          BEQL    6$
                 05        50  A3     6442  E1 00093          BBC     (PTR)[IDX], 80(LT), 5$             ; 2736
                                       50  DD 00099          PUSHL   R0                                 ; 2738
                                68     01  FB 0009B          CALLS   #1, LIB$PUT_OUTPUT
        FFE4     52             02 00000040  8F  F1 0009E 5$:   ACBL    #64, #2, IDX, 4$                   ; 2736
                                       04 000A8 6$:      RET                                        ; 2744

; Routine Size:  169 bytes,    Routine Base:  $CODE$ + 0BD7

; 2631          2745  1
```

```
%SBTTL  'LCP_SHOWDEVCOU  Show Ethernet device counters'
ROUTINE LCP_SHOWDEVCOU : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Convert and write the Ethernet device counters to sys$output.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       counters in LAT driver
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LITERAL
        CTRBUFSIZE = 512
        ;

    LOCAL
        CTRDSC : BLOCK [DSC$K_Z_BLN, BYTE],
        STATUS
        ;

    !
    !  Setup the counters return buffer descriptor
    !
    $SETDSC (CTRDSC, CTRBUFSIZE, AREA);

    !
    !  Read and display the counters
    !
    IF  NOT (STATUS = $CMKRNL (ROUTIN = LCPK_SHOWDEVCOU) )
        OR
        NOT (STATUS = LCP_DISPDEVCOU (CTRDSC) )
    THEN
        SIGNAL (.STATUS)
    ;
```

```
; 2690          2803  1     END;


                                        0000 00000 LCP_SHOWDEVCOU:
                                                        .WORD   Save nothing                    ; 2747
                        5E              04  C2 00002    SUBL2   #4, SP                          ;
                        7E      0200    8F  3C 00005    MOVZWL  #512, CTRDSC                    ; 2791
                 04     AE      0000'   CF  9E 0000A    MOVAB   AREA, CTRDSC+4                  ;
                                        7E  D4 00010    CLRL    -(SP)                           ; 2796
                                0000V   CF  9F 00012    PUSHAB  LCPK_SHOWDEVCOU                 ;
             00000000G  00              02  FB 00016    CALLS   #2, SYS$CMKRNL                  ;
                        0A              50  E9 0001D    BLBC    STATUS, 1$                      ;
                                        5E  DD 00020    PUSHL   SP                              ; 2798
                0000V   CF              01  FB 00022    CALLS   #1, LCP_DISPDEVCOU              ;
                        09              50  E8 00027    BLBS    STATUS, 2$                      ;
                                        50  DD 0002A 1$:  PUSHL  STATUS                         ; 2800
             00000000G  00              01  FB 0002C    CALLS   #1, LIB$SIGNAL                  ;
                                            04 00033 2$:  RET                                   ; 2803

; Routine Size:  52 bytes,    Routine Base:  $CODE$ + 0C80
```

```
%SBTTL  'LCPK_SHOWDEVCOU  Show Ethernet device counters'
ROUTINE LCPK_SHOWDEVCOU (CTRDSC) =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Convert and write the Ethernet device counters to sys$output.
!
! FORMAL PARAMETERS:
!
!       CTRDSC          Address of counter block descriptor
!
! IMPLICIT INPUTS:
!
!       counters in Ethernet driver
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       error values returned
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LITERAL
        DEVNAMSIZE = 16,
        CTRBUFSIZE = 512
        ;

    BIND
        DEVNAMCTRL = DESCRIPTOR ('!AC!UW')
        ;

    LOCAL
        LT : REF BLOCK [,BYTE],
        DEVNAMSTR : VECTOR [DEVNAMSIZE, BYTE],
        DEVNAMDSC : BLOCK [DSC$K_Z_BLN, BYTE],
        RDCTRDSC : BLOCK [DSC$K_Z_BLN, BYTE],
        UCB : REF BLOCK [,BYTE],
        ORB : REF BLOCK [,BYTE],
        DDB : REF BLOCK [,BYTE],
        IOSB : VECTOR [4, WORD],
        STATUS,
        DEVCHAN : WORD,
        PID,
        OWNUIC,
        VPROT : WORD
        ;
```

```
; 2749          2861  2
; 2750          2862  2      !
; 2751          2863  2      ! If the data link UCB is gone, then we are inactive
; 2752          2864  2      !
; 2753          2865  2      LT = .LCP$L_AREA;
; 2754          2866  2      IF (UCB = .LT [GHB$L_UCB] ) EQL 0
; 2755          2867  2      THEN
; 2756          2868  2          RETURN SS$_DEVINACT
; 2757          2869  2      ;
; 2758          2870  2      !
; 2759          2871  2      ! Get pointers to the ucb and ddb for datalink to make its name
; 2760          2872  2      ! so we can assign a channel to the data link device ucb
; 2761          2873  2      !
; 2762          2874  2      DDB = .UCB [UCB$L_DDB];
; 2763          2875  2      DEVNAMDSC [DSC$A_POINTER] = DEVNAMSTR;
; 2764          2876  2      DEVNAMDSC [DSC$W_LENGTH] = DEVNAMSIZE;
; 2765        P 2877  2      STATUS = $FAO
; 2766        P 2878  2          (
; 2767        P 2879  2          DEVNAMCTRL,
; 2768        P 2880  2          DEVNAMDSC [DSC$W_LENGTH],
; 2769        P 2881  2          DEVNAMDSC,
; 2770        P 2882  2          DDB [DDB$T_NAME],
; 2771        P 2883  2          .UCB [UCB$W_UNIT]
; 2772          2884  2          );
; 2773          2885  2      IF NOT .STATUS
; 2774          2886  2      THEN
; 2775          2887  2          RETURN .STATUS
; 2776          2888  2      ;
; 2777          2889  2      !
; 2778          2890  2      !  Fix the ucb so we can assign a channel to it... save current
; 2779          2891  2      !  settings first.
; 2780          2892  2      !
; 2781          2893  2      PID = .UCB [UCB$L_PID];
; 2782          2894  2      ORB = .UCB[UCB$L_ORB];
; 2783          2895  2      OWNUIC = .ORB [ORB$L_OWNER];
; 2784          2896  2      VPROT = .ORB [ORB$W_PROT];
; 2785          2897  2      UCB [UCB$L_PID] = 0;
; 2786          2898  2      ORB [ORB$L_OWNER] = 0;
; 2787          2899  2      ORB [ORB$W_PROT] = 0;
; 2788          2900  2      !
; 2789          2901  2      ! Assign a channel to hold it here and allow us to shut it down
; 2790          2902  2      ! later and deassign the ucb to cause it to go away.
; 2791          2903  2      !
; 2792          2904  2      STATUS = $ASSIGN (DEVNAM = DEVNAMDSC, CHAN = DEVCHAN);
; 2793          2905  2      IF NOT .STATUS
; 2794          2906  2      THEN
; 2795          2907  2          RETURN .STATUS
; 2796          2908  2      ;
; 2797          2909  2
; 2798          2910  2      !
; 2799          2911  2      !  Setup the counters return buffer descriptor
; 2800          2912  2      !
; 2801          2913  2      $SETDSC (RDCTRDSC, CTRBUFSIZE, AREA);
; 2802          2914  2
; 2803          2915  2      !
; 2804          2916  2      !  Read the counters from the Ethernet device
; 2805          2917  2      !
```

```
; 2806        P 2918  2     STATUS = $QIOW
; 2807        P 2919  2         (
; 2808        P 2920  2         FUNC = IO$_SENSEMODE OR IO$M_CTRL OR IO$M_RD_COUNT,
; 2809        P 2921  2         CHAN = .DEVCHAN,
; 2810        P 2922  2         IOSB = IOSB,
; 2811        P 2923  2         P2 = RDCTRDSC
; 2812          2924  2         );
; 2813          2925  2
; 2814          2926  2     !
; 2815          2927  2     !  Restore device settings
; 2816          2928  2     !
; 2817          2929  2     UCB [UCB$L_PID] = .PID;
; 2818          2930  2     ORB [ORB$L_OWNER] = .OWNUIC;
; 2819          2931  2     ORB [ORB$W_PROT] = .VPROT;
; 2820          2932  2
; 2821          2933  2     !
; 2822          2934  2     ! Deassign the channel that we have.
; 2823          2935  2     !
; 2824          2936  2     $DASSGN (CHAN = .DEVCHAN);
; 2825          2937  2
; 2826          2938  2     RETURN .STATUS              ! Return the status from the READ QIO
; 2827          2939  2
; 2828          2940  1     END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                        57  55  21  43  41  21  00E00 P.AFP:  .ASCII  \!AC!UW\                          ;
                                                00E06         .BLKB   2
                                    00000006    00E08 P.AFO:  .LONG   6                                  ;
                                    00000000'   00E0C         .ADDRESS P.AFP                           ;

                                                      DEVNAMCTRL=        P.AFO


                                                        .PSECT  $CODE$,NOWRT,2

                                        00FC 00000 LCPK_SHOWDEVCOU:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7                   ; 2805
                 5E                 2C  C2 00002        SUBL2   #44, SP                                  ;
                 50        0000'    CF  D0 00005        MOVL    LCP$L_AREA, LT                           ; 2865
                 53           14    A0  D0 0000A        MOVL    20(LT), UCB                              ; 2866
                                    06  12 0000E        BNEQ    1$                                       ;
                 50         20D4    8F  3C 00010        MOVZWL  #8404, R0                                ; 2868
                                        04 00015        RET                                              ;
                 50           28    A3  D0 00016 1$:    MOVL    40(UCB), DDB                             ; 2874
            18   AE           1C    AE  9E 0001A        MOVAB   DEVNAMSTR, DEVNAMDSC+4                   ; 2875
            14   AE                 10  B0 0001F        MOVW    #16, DEVNAMDSC                           ; 2876
                 7E           54    A3  3C 00023        MOVZWL  84(UCB), -(SP)                           ; 2884
                              14    A0  9F 00027        PUSHAB  20(DDB)                                  ;
                              1C    AE  9F 0002A        PUSHAB  DEVNAMDSC                                ;
                              20    AE  9F 0002D        PUSHAB  DEVNAMDSC                                ;
                         0000'      CF  9F 00030        PUSHAB  DEVNAMCTRL                               ;
      00000000G  00                 05  FB 00034        CALLS   #5, SYS$FAO                              ;
                 54                 50  D0 0003B        MOVL    R0, STATUS                               ;
                 70                 54  E9 0003E        BLBC    STATUS, 2$                               ; 2885
```

```
                                57      2C   A3  D0 00041          MOVL    44(UCB), PID                          ; 2893
                                52      1C   A3  D0 00045          MOVL    28(UCB), ORB                          ; 2894
                                56           62  D0 00049          MOVL    (ORB), OWNUIC                         ; 2895
                                55      18   A2  B0 0004C          MOVW    24(ORB), VPROT                        ; 2896
                                        2C   A3  D4 00050          CLRL    44(UCB)                               ; 2897
                                             62  D4 00053          CLRL    (ORB)                                 ; 2898
                                        18   A2  B4 00055          CLRW    24(ORB)                               ; 2899
                                             7E  7C 00058          CLRQ    -(SP)                                 ; 2904
                                        08   AE  9F 0005A          PUSHAB  DEVCHAN                               ;
                                        20   AE  9F 0005D          PUSHAB  DEVNAMDSC                             ;
                        00000000G       00       04  FB 00060      CALLS   #4, SYS$ASSIGN                        ;
                                        54       50  D0 00067      MOVL    R0, STATUS                            ;
                                        44       54  E9 0006A      BLBC    STATUS, 2$                            ; 2905
                                0C      AE  0200 8F  3C 0006D      MOVZWL  #512, RDCTRDSC                        ; 2913
                                10      AE  0000'CF  9E 00073      MOVAB   AREA, RDCTRDSC+4                      ;
                                                 7E  7C 00079      CLRQ    -(SP)                                 ; 2924
                                                 7E  7C 0007B      CLRQ    -(SP)                                 ;
                                        1C   AE  9F 0007D          PUSHAB  RDCTRDSC                              ;
                                                 7E  7C 00080      CLRQ    -(SP)                                 ;
                                                 7E  D4 00082      CLRL    -(SP)                                 ;
                                        24   AE  9F 00084          PUSHAB  IOSB                                  ;
                                7E      0327 8F  3C 00087          MOVZWL  #807, -(SP)                           ;
                                7E      28   AE  3C 0008C          MOVZWL  DEVCHAN, -(SP)                        ;
                                                 7E  D4 00090      CLRL    -(SP)                                 ;
                        00000000G       00       0C  FB 00092      CALLS   #12, SYS$QIOW                         ;
                                        54       50  D0 00099      MOVL    R0, STATUS                            ;
                                2C      A3       57  D0 0009C      MOVL    PID, 44(UCB)                          ; 2929
                                        62       56  D0 000A0      MOVL    OWNUIC, (ORB)                         ; 2930
                                18      A2       55  B0 000A3      MOVW    VPROT, 24(ORB)                        ; 2931
                                        7E       6E  3C 000A7      MOVZWL  DEVCHAN, -(SP)                        ; 2936
                        00000000G       00       01  FB 000AA      CALLS   #1, SYS$DASSGN                        ;
                                        50       54  D0 000B1 2$:  MOVL    STATUS, R0                            ; 2938
                                                     04 000B4      RET                                           ; 2940

; Routine Size:  181 bytes,    Routine Base:  $CODE$ + 0CB4
```

```
%SBTTL  'LCP_DISPDEVCOU  Display the Ethernet device counters'
ROUTINE LCP_DISPDEVCOU (CTRDSC) =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Convert and write the all the Ethernet device counters to sys$output.
!
! FORMAL PARAMETERS:
!
!       CTRDSC          Address of descriptor of the counters string
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       error values returned
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LITERAL PARMSIZE = 50;

    LOCAL
        PARAM : VECTOR [PARMSIZE, LONG],
        PINDX,
        CTRID,
        STATUS,
        VALUE,
        BITMAP,
        RCV_BITMAP : BLOCK [4, BYTE],
        XMT_BITMAP : BLOCK [4, BYTE],
        PTR : REF VECTOR []
        ;

    RCV_BITMAP = 0;
    XMT_BITMAP = 0;
    !
    !  Loop through all the counters
    !
    PINDX = 0;
    CTRID = CTRNAMES;
    WHILE ..CTRID NEQ -1
    DO
        BEGIN
```

```
; 2887          3  !
; 2888          3  !  Get the counter values
; 2889          3  !
; 2890          3  LCP_FINDCOU (.CTRDSC, ..CTRID, VALUE, BITMAP);
; 2891          3  PARAM [.PINDX] = .VALUE;
; 2892          3  SELECTONEU ..CTRID OF
; 2893          3  SET
; 2894          3
; 2895          3  [NMA$C_CTLIN_RFL] :                    ! If receive failures
; 2896          4      (
; 2897          4      RCV_BITMAP = .BITMAP               ! Save the bitmap
; 2898          3      );
; 2899          3
; 2900          3  [NMA$C_CTLIN_SFL] :                    ! Is send failures
; 2901          4      (
; 2902          4      XMT_BITMAP = .BITMAP               ! Save the bitmap
; 2903          3      );
; 2904          3
; 2905          3  TES
; 2906          3  ;
; 2907          3
; 2908          3  PINDX = .PINDX + 1;                    ! Skip to next parameter
; 2909          3  CTRID = .CTRID + 4                     ! Skip to next counter ID
; 2910          3
; 2911          3  END
; 2912          2 ;
; 2913          2
; 2914          2 !
; 2915          2 !  Convert the counters for output
; 2916          2 !
; 2917          2 $SETDSC (OUTDSC, OUTBUFSIZE, OUTBUF);
; 2918          2
; 2919        P 2 STATUS = $FAOL
; 2920        P 2     (
; 2921        P 2     CTRSTR = DEVCTRSTR,
; 2922        P 2     OUTLEN = OUTDSC [DSC$W_LENGTH],
; 2923        P 2     OUTBUF = OUTDSC,
; 2924        P 2     PRMLST = PARAM[0]
; 2925          2     );
; 2926          2
; 2927          2 IF NOT .STATUS THEN RETURN .STATUS;
; 2928          2
; 2929          2 STATUS = LIB$PUT_OUTPUT (OUTDSC);
; 2930          2
; 2931          2 IF NOT .STATUS THEN RETURN .STATUS;
; 2932          2
; 2933          2 !
; 2934          2 ! Print the receive mask bit names that are present
; 2935          2 !
; 2936          2 IF .RCV_BITMAP [0, 0, 16, 0] NEQ 0
; 2937          2 THEN
; 2938          3     BEGIN
; 2939          3     !
; 2940          3     !  Print the header
; 2941          3     !
; 2942          3     STATUS = LIB$PUT_OUTPUT (RCVERRMASK);
; 2943          3     PTR = RCVERRBITS;                   ! Point to the translation tbl
```

```
: 2944   3055  3          INCR INDX FROM 0 TO 32 BY 2
: 2945   3056  3          DO
: 2946   3057  4              BEGIN
: 2947   3058  4              !
: 2948   3059  4              ! If the string desc is blank, then exit loop
: 2949   3060  4              !
: 2950   3061  4              IF .PTR [.INDX + 1] EQL 0
: 2951   3062  4              THEN
: 2952   3063  4                  EXITLOOP
: 2953   3064  4              ;
: 2954   3065  4              !
: 2955   3066  4              ! If the bit is set, then write the string
: 2956   3067  4              !
: 2957   3068  4              IF .RCV_BITMAP [0, .PTR [.INDX], 1, 0]
: 2958   3069  4              THEN
: 2959   3070  4                  LIB$PUT_OUTPUT (.PTR [.INDX + 1] )
: 2960   3071  4              END
: 2961   3072  3          ;
: 2962   3073  3          END
: 2963   3074  2      ;
: 2964   3075  2
: 2965   3076  2      !
: 2966   3077  2      ! Print the transmit mask bit names that are present
: 2967   3078  2      !
: 2968   3079  2      IF .XMT_BITMAP [0, 0, 16, 0] NEQ 0
: 2969   3080  2      THEN
: 2970   3081  3          BEGIN
: 2971   3082  3          !
: 2972   3083  3          !  Print the header
: 2973   3084  3          !
: 2974   3085  3          STATUS = LIB$PUT_OUTPUT (XMTERRMASK);
: 2975   3086  3          PTR = XMTERRBITS;                    ! Point to the translation tbl
: 2976   3087  3          INCR INDX FROM 0 TO 32 BY 2
: 2977   3088  3          DO
: 2978   3089  4              BEGIN
: 2979   3090  4              !
: 2980   3091  4              ! If the string desc is blank, then exit loop
: 2981   3092  4              !
: 2982   3093  4              IF .PTR [.INDX + 1] EQL 0
: 2983   3094  4              THEN
: 2984   3095  4                  EXITLOOP
: 2985   3096  4              ;
: 2986   3097  4              !
: 2987   3098  4              ! If the bit is set, then write the string
: 2988   3099  4              !
: 2989   3100  4              IF .XMT_BITMAP [0, .PTR [.INDX], 1, 0]
: 2990   3101  4              THEN
: 2991   3102  4                  LIB$PUT_OUTPUT (.PTR [.INDX + 1] )
: 2992   3103  4              END
: 2993   3104  3          ;
: 2994   3105  3          END
: 2995   3106  2      ;
: 2996   3107  2
: 2997   3108  2      SUCCESS
: 2998   3109  2
: 2999   3110  1      END;
```

```
                                    01FC 00000 LCP_DISPDEVCOU:
                                                       .WORD   Save R2,R3,R4,R5,R6,R7,R8                  ; 2942
                 58      0000'  CF  9E 00002          MOVAB   OUTDSC, R8                                 ;
                 57 00000000G   00  9E 00007          MOVAB   LIB$PUT_OUTPUT, R7                         ;
                 5E      FF30   CE  9E 0000E          MOVAB   -208(SP), SP                               ;
                                55  7C 00013          CLRQ    XMT_BITMAP                                 ; 2989
                                52  D4 00015          CLRL    PINDX                                      ; 2993
                 53      0000'  CF  9E 00017          MOVAB   CTRNAMES, CTRID                            ; 2994
        FFFFFFFF  8F            63  D1 0001C 1$:      CMPL    (CTRID), #-1                               ; 2995
                                36  13 00023          BEQL    4$                                         ;
                                5E  DD 00025          PUSHL   SP                                         ; 3001
                           08   AE  9F 00027          PUSHAB  VALUE                                      ;
                                63  DD 0002A          PUSHL   (CTRID)                                    ;
                           04   AC  DD 0002C          PUSHL   CTRDSC                                     ;
            0000V  CF           04  FB 0002F          CALLS   #4, LCP_FINDCOU                            ;
              08 AE42           04  AE  D0 00034      MOVL    VALUE, PARAM[PINDX]                        ; 3002
        00000426  8F            63  D1 0003A          CMPL    (CTRID), #1062                             ; 3006
                                05  12 00041          BNEQ    2$                                         ;
                 56             6E  D0 00043          MOVL    BITMAP, RCV_BITMAP                         ; 3008
                                0C  11 00046          BRB     3$                                         ; 3007
        00000424  8F            63  D1 00048 2$:      CMPL    (CTRID), #1060                             ; 3011
                                03  12 0004F          BNEQ    3$                                         ;
                 55             6E  D0 00051          MOVL    BITMAP, XMT_BITMAP                         ; 3013
                                52  D6 00054 3$:      INCL    PINDX                                      ; 3019
                 53             04  C0 00056          ADDL2   #4, CTRID                                  ; 3020
                                C1  11 00059          BRB     1$                                         ;
                 68      0400   8F  3C 0005B 4$:      MOVZWL  #1024, OUTDSC                              ; 3028
              04 A8         08  A8  9E 00060          MOVAB   OUTBUF, OUTDSC+4                           ;
                            08  AE  9F 00065          PUSHAB  PARAM                                      ; 3036
                                58  DD 00068          PUSHL   R8                                         ;
                                58  DD 0006A          PUSHL   R8                                         ;
                         0000'  CF  9F 0006C          PUSHAB  DEVCTRSTR                                  ;
     00000000G  00              04  FB 00070          CALLS   #4, SYS$FAOL                               ;
                 54             50  D0 00077          MOVL    R0, STATUS                                 ;
                 0B             54  E9 0007A          BLBC    STATUS, 5$                                 ; 3038
                                58  DD 0007D          PUSHL   R8                                         ; 3040
                 67             01  FB 0007F          CALLS   #1, LIB$PUT_OUTPUT                         ;
                 54             50  D0 00082          MOVL    R0, STATUS                                 ;
                 04             54  E8 00085          BLBS    STATUS, 6$                                 ; 3042
                 50             54  D0 00088 5$:      MOVL    STATUS, R0                                 ;
                                    04 0008B          RET                                                ;
                                56  B5 0008C 6$:      TSTW    RCV_BITMAP                                 ; 3047
                                28  13 0008E          BEQL    9$                                         ;
                         0000'  CF  9F 00090          PUSHAB  RCVERRMASK                                 ; 3053
                 67             01  FB 00094          CALLS   #1, LIB$PUT_OUTPUT                         ;
                 54             50  D0 00097          MOVL    R0, STATUS                                 ;
                 53      0000'  CF  9E 0009A          MOVAB   RCVERRBITS, PTR                            ; 3054
                                52  D4 0009F          CLRL    INDX                                       ; 3061
                 50        04 A342  D0 000A1 7$:      MOVL    4(PTR)[INDX], R0                           ;
                                10  13 000A6          BEQL    9$                                         ;
     05          56          6342  E1 000A8          BBC     (PTR)[INDX], RCV_BITMAP, 8$                ; 3068
                                50  DD 000AD          PUSHL   R0                                         ; 3070
                 67             01  FB 000AF          CALLS   #1, LIB$PUT_OUTPUT                         ;
```

```
        FFE9        52          02          20  F1 000B2 8$:    ACBL    #32, #2, INDX, 7$                    ; 3068
                                            55  B5 000B8 9$:    TSTW    XMT_BITMAP                           ; 3079
                                            28  13 000BA        BEQL    12$                                  ;
                                  0000'     CF  9F 000BC        PUSHAB  XMTERRMASK                           ; 3085
                                67          01  FB 000C0        CALLS   #1, LIB$PUT_OUTPUT                   ;
                                54          50  D0 000C3        MOVL    R0, STATUS                           ;
                                53        0000' CF  9E 000C6    MOVAB   XMTERRBITS, PTR                      ; 3086
                                            54  D4 000CB        CLRL    INDX                                 ; 3093
                                50        04 A344 D0 000CD 10$:  MOVL    4(PTR)[INDX], R0                     ;
                                            10  13 000D2        BEQL    12$                                  ;
                    05          55        6344  E1 000D4        BBC     (PTR)[INDX], XMT_BITMAP, 11$         ; 3100
                                            50  DD 000D9        PUSHL   R0                                   ; 3102
                                67          01  FB 000DB        CALLS   #1, LIB$PUT_OUTPUT                   ;
        FFE9        54          02          20  F1 000DE 11$:   ACBL    #32, #2, INDX, 10$                   ; 3100
                                50          01  D0 000E4 12$:   MOVL    #1, R0                               ; 3110
                                                04 000E7        RET                                          ;

; Routine Size:  232 bytes,    Routine Base:  $CODE$ + 0D69
```

```
3111  1 %SBTTL  'LCP_FINDCOU  Find the specific device counter'
3112  1 ROUTINE LCP_FINDCOU (CTRDSC, CTRNAME, VALUE, BITMAP) =
3113  1
3114  1 !++
3115  1 ! FUNCTIONAL DESCRIPTION:
3116  1 !
3117  1 !   Convert and write the all the Ethernet device counters to sys$output.
3118  1 !
3119  1 ! FORMAL PARAMETERS:
3120  1 !
3121  1 !       CTRDSC          Address of descriptor of the counters string
3122  1 !       CTRNAME         Name (ID) of the counter id code to search for
3123  1 !       VALUE           Returned longword value for the counter if found
3124  1 !       BITMAP          Returned longword value for the bitmap if found
3125  1 !
3126  1 ! IMPLICIT INPUTS:
3127  1 !
3128  1 !       NONE
3129  1 !
3130  1 ! IMPLICIT OUTPUTS:
3131  1 !
3132  1 !       NONE
3133  1 !
3134  1 ! ROUTINE VALUE:
3135  1 ! COMPLETION CODES:
3136  1 !
3137  1 !       error values returned
3138  1 !
3139  1 ! SIDE EFFECTS:
3140  1 !
3141  1 !       NONE
3142  1 !
3143  1 !--
3144  1
3145  2     BEGIN
3146  2
3147  2     MAP
3148  2         CTRDSC : REF BLOCK [,BYTE]
3149  2         ;
3150  2
3151  2     LOCAL
3152  2         CTRLEN,
3153  2         CTRPTR : REF BLOCK [,BYTE],
3154  2         STATUS,
3155  2         COUCOD,
3156  2         COUVAL,
3157  2         COUID : BLOCK [4, BYTE],
3158  2         ERRMAP
3159  2         ;
3160  2
3161  2     CTRPTR = .CTRDSC [DSC$A_POINTER];
3162  2     CTRLEN = .CTRDSC [DSC$W_LENGTH];
3163  2     COUCOD = -1;                                  ! Set to illegal value
3164  2     WHILE .CTRLEN GTR 0 AND
3165  2           .COUCOD NEQ .CTRNAME
3166  2     DO
3167  3         BEGIN
```

```
3168  3          COUID = .CTRPTR [0, 0 , 16, 0];          ! Get the counter id + bits
3169  3          COUCOD = .COUID [NMA$V_CNT_TYP];         ! Get just the counter code
3170  3          CTRPTR = .CTRPTR + 2;                    ! Skip counter id field
3171  3          CTRLEN = .CTRLEN - 2;                    ! Consume byte count
3172  3
3173  3          !
3174  3          !  Get the bitmap if present
3175  3          !
3176  3          IF      .D [NMA$V_CNT_MAP]
3177  3          THEN
3178  4              BEGIN
3179  4              ERRMAP = .CTRPTR [0, 0, 16, 0];      ! Get the bitmap
3180  4              CTRPTR = .CTRPTR + 2;                ! Skip it
3181  4              CTRLEN = .CTRLEN - 2                 ! Consume bytes
3182  4              END
3183  3          ELSE
3184  3              ERRMAP = 0
3185  3          ;
3186  3
3187  3          !
3188  3          !  Dispatch on counter width
3189  3          !
3190  3          SELECTONEU .COUID [NMA$V_CNT_WID] OF
3191  3          SET
3192  3
3193  3          [0] :                                    ! Unknown???
3194  4              (
3195  5              RETURN (LCP$_INTERNAL)
3196  3              );
3197  3
3198  3          [1] :                                    ! 8 bit counter
3199  4              (
3200  4              COUVAL = .CTRPTR [0, 0, 8, 0];
3201  4              CTRPTR = .CTRPTR + 1
3202  3              );
3203  3
3204  3          [2] :                                    ! 16 bit counter
3205  4              (
3206  4              COUVAL = .CTRPTR [0, 0, 16, 0];
3207  4              CTRPTR = .CTRPTR + 2
3208  3              );
3209  3
3210  3          [3] :                                    ! 32 bit counter
3211  4              (
3212  4              COUVAL = .CTRPTR [0, 0, 32, 0];
3213  4              CTRPTR = .CTRPTR + 4
3214  3              );
3215  3
3216  3          TES
3217  3          ;
3218  3
3219  3          END
3220  2      ;
3221  2
3222  2      IF .COUCOD NEQ .CTRNAME
3223  2      THEN
3224  3          BEGIN
```

```
; 3115   3225  3          .VALUE = 0;
; 3116   3226  3          .BITMAP = 0;
; 3117   3227  3          STATUS = FAILURE
; 3118   3228  3          END
; 3119   3229  2      ELSE
; 3120   3230  3          BEGIN
; 3121   3231  3          .VALUE = .COUVAL;
; 3122   3232  3          .BITMAP = .ERRMAP;
; 3123   3233  3          STATUS = SUCCESS
; 3124   3234  3          END
; 3125   3235  2      ;
; 3126   3236  2
; 3127   3237  2      RETURN .STATUS
; 3128   3238  2
; 3129   3239  1      END;
```

```
                                    003C 00000 LCP_FINDCOU:
                                                .WORD   Save R2,R3,R4,R5                        ; 3112
                50      04  AC  D0 00002        MOVL    CTRDSC, R0                              ; 3161
                51      04  A0  D0 00006        MOVL    4(R0), CTRPTR                           ;
                54          60  3C 0000A        MOVZWL  (R0), CTRLEN                            ; 3162
                53          01  CE 0000D        MNEGL   #1, COUCOD                              ; 3163
                            54  D5 00010 1$:    TSTL    CTRLEN                                  ; 3164
                            52  15 00012        BLEQ    7$                                      ;
            08  AC          53  D1 00014        CMPL    COUCOD, CTRNAME                         ; 3165
                            4C  13 00018        BEQL    7$                                      ;
                50          81  3C 0001A        MOVZWL  (CTRPTR)+, COUID                        ; 3168
53      50      0C          00  EF 0001D        EXTZV   #0, #12, COUID, COUCOD                  ; 3169
                54          02  C2 00022        SUBL2   #2, CTRLEN                              ; 3171
        08      50          0C  E1 00025        BBC     #12, COUID, 2$                          ; 3176
                55          81  3C 00029        MOVZWL  (CTRPTR)+, ERRMAP                       ; 3179
                54          02  C2 0002C        SUBL2   #2, CTRLEN                              ; 3181
                            02  11 0002F        BRB     3$                                      ;
                            55  D4 00031 2$:    CLRL    ERRMAP                                  ; 3184
            6000    8F      50  B3 00033 3$:    BITW    COUID, #24576                           ; 3193
                            08  12 00038        BNEQ    4$                                      ;
                50 00000000G 8F D0 0003A        MOVL    #LCP$_INTERNAL, R0                      ; 3195
                            04  00041           RET                                             ;
01      50      02          0D  ED 00042 4$:    CMPZV   #13, #2, COUID, #1                      ; 3198
                            05  12 00047        BNEQ    5$                                      ;
                52          81  9A 00049        MOVZBL  (CTRPTR)+, COUVAL                       ; 3200
                            C2  11 0004C        BRB     1$                                      ; 3201
02      50      02          0D  ED 0004E 5$:    CMPZV   #13, #2, COUID, #2                      ; 3204
                            05  12 00053        BNEQ    6$                                      ;
                52          81  3C 00055        MOVZWL  (CTRPTR)+, COUVAL                       ; 3206
                            B6  11 00058        BRB     1$                                      ; 3207
03      50      02          0D  ED 0005A 6$:    CMPZV   #13, #2, COUID, #3                      ; 3210
                            AF  12 0005F        BNEQ    1$                                      ;
                52          81  D0 00061        MOVL    (CTRPTR)+, COUVAL                       ; 3212
                            AA  11 00064        BRB     1$                                      ; 3164
            08  AC          53  D1 00066 7$:    CMPL    COUCOD, CTRNAME                         ; 3222
                            09  13 0006A        BEQL    8$                                      ;
                        0C  BC  D4 0006C        CLRL    @VALUE                                  ; 3225
```

```
                                    10   BC  D4 0006F          CLRL    @BITMAP                          ; 3226
                                         50  D4 00072          CLRL    STATUS                           ; 3227
                                             04 00074          RET                                      ;
                        0C   BC          52  D0 00075 8$:      MOVL    COUVAL, @VALUE                   ; 3231
                        10   BC          55  D0 00079          MOVL    ERRMAP, @BITMAP                  ; 3232
                             50          01  D0 0007D          MOVL    #1, STATUS                       ; 3233
                                             04 00080          RET                                      ; 3239

; Routine Size:  129 bytes,    Routine Base:  $CODE$ + 0E51
```

```
%SBTTL  'LCP_SHOWSERVCOU  Show SERVER counters'
ROUTINE LCP_SHOWSERVCOU : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Convert and write the all the SERVER counters to sys$output.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       counters in LAT driver
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    BIND
        UNKNOWN = DESCRIPTOR ('UNKNOWN')
        ;

    LITERAL
        BUFSIZE = 24
        ;

    LOCAL
        DSC : BLOCK [DSC$K_Z_BLN, BYTE],
        CTRBUF : VECTOR [BUFSIZE],
        PTR : REF BLOCK [,BYTE],
        STATUS
        ;

    CH$FILL (255, GHB$S_COMM_AREA, AREA);

    !
    !  Get the counters if we can, and if not then signal the error
    !
    IF NOT
        (STATUS = $CMKRNL (ROUTIN = LCPK_GETSERVCTRS) )
    THEN
```

```
3188          2          SIGNAL (.STATUS)
3189          2      ;
3190          2
3191          2      !
3192          2      !  Convert the counters for output one at a time
3193          2      !
3194          2      IF .AREA_RET_SIZE EQL 0
3195          2      THEN
3196          2          SIGNAL (LCP$_NOSERVERS)
3197          2      ELSE
3198          3          BEGIN
3199          3          INCR INDX FROM AREA
3200          4                  TO AREA + .AREA_RET_SIZE - (LCB_K_LENGTH + GHB$K_NAMELEN + 1)
3201          3                  BY LCB_K_LENGTH + GHB$K_NAMELEN + 1
3202          3          DO
3203          4              BEGIN
3204          4              PTR = .INDX;
3205          4              CTRBUF [1] = .PTR [LCB_L_MSG_RCV];
3206          4              CTRBUF [2] = .PTR [LCB_L_MSG_XMT];
3207          4              CTRBUF [3] = .PTR [LCB_L_MSG_REXMT];
3208          4              CTRBUF [4] = .PTR [LCB_W_SEQ_ERR];
3209          4              CTRBUF [5] = .PTR [LCB_B_INV_MSG];
3210          4              CTRBUF [6] = .PTR [LCB_B_INV_SLOT];
3211          4
3212          4              PTR = .PTR + LCB_K_LENGTH;          ! Skip over counters
3213          4
3214          4              IF .PTR [0, 0, 8, 0] EQL 0
3215          4              THEN
3216          4                  CTRBUF [0] = UNKNOWN
3217          4              ELSE
3218          5                  BEGIN
3219          5                  CTRBUF [0] = DSC;
3220          6                  $SETDSC (DSC, .PTR [0, 0, 8, 0], PTR [1, 0, 8, 0])
3221          5                  END
3222          4              ;
3223          4
3224          4              $SETDSC (OUTDSC, OUTBUFSIZE, OUTBUF);
3225          4
3226        P 4              STATUS = $FAOL
3227        P 4                  (
3228        P 4                  CTRSTR = SERVCTRSTR,
3229        P 4                  OUTLEN = OUTDSC [DSC$W_LENGTH],
3230        P 4                  OUTBUF = OUTDSC,
3231        P 4                  PRMLST = CTRBUF [0]
3232          4                  );
3233          4
3234          4              IF NOT .STATUS THEN SIGNAL (.STATUS);
3235          4
3236          4              STATUS = LIB$PUT_OUTPUT (OUTDSC);
3237          4
3238          4              IF NOT .STATUS THEN SIGNAL (.STATUS)
3239          4
3240          4              END
3241          3          ;
3242          3
3243          3          END
3244          3
```

```
; 3245          3354  1     END;


                                                                  .PSECT  $PLIT$,NOWRT,NOEXE,2

                        4E  57  4F  4E  4B  4E  55  00E10 P.AFR:  .ASCII  \UNKNOWN\                          ;
                                                    00E17         .BLKB   1
                                        00000007    00E18 P.AFQ:  .LONG   7                                  ;
                                        00000000'   00E1C         .ADDRESS P.AFR                           ;

                                                          UNKNOWN=          P.AFQ


                                                                  .PSECT  $CODE$,NOWRT,2

                                                 00FC 00000 LCP_SHOWSERVCOU:
                                                                  .WORD   Save R2,R3,R4,R5,R6,R7           ; 3241
                                 57 00000000G   00   9E 00002     MOVAB   LIB$SIGNAL, R7                   ;
                                 56     0000'   CF   9E 00009     MOVAB   OUTDSC, R6                       ;
                                 5E        98   AE   9E 0000E     MOVAB   -104(SP), SP                     ;
    0060  8F        FF  8F       6E            00   2C 00012     MOVC5   #0, (SP), #255, #96, AREA        ; 3289
                                        F5FC   C6      0001A                                             ;
                                               7E   D4 0001D     CLRL    -(SP)                            ; 3295
                                        0000V  CF   9F 0001F     PUSHAB  LCPK_GETSERVCTRS                 ;
                          00000000G   00       02   FB 00023     CALLS   #2, SYS$CMKRNL                   ;
                                 53            50   D0 0002A     MOVL    R0, STATUS                       ;
                                 05            53   E8 0002D     BLBS    STATUS, 1$                       ;
                                               53   DD 00030     PUSHL   STATUS                           ; 3297
                                 67            01   FB 00032     CALLS   #1, LIB$SIGNAL                   ;
                                 52        FC  A6   D0 00035 1$: MOVL    AREA_RET_SIZE, R2                ; 3303
                                               0A   12 00039     BNEQ    2$                               ;
                              00000000G        8F   DD 0003B     PUSHL   #LCP$_NOSERVERS                  ; 3305
                                 67            01   FB 00041     CALLS   #1, LIB$SIGNAL                   ;
                                               04   00044     RET                                      ;
                                 54      F5FC  C6   9E 00045 2$: MOVAB   AREA, R4                         ; 3308
                                 55      F5DB C642  9E 0004A     MOVAB   AREA-33[R2], R5                  ; 3309
                                               7B   11 00050     BRB     8$                               ; 3310
                                 52            54   D0 00052 3$: MOVL    INDX, PTR                        ; 3313
    04  AE                    04   A2   D0 00055     MOVL    4(PTR), CTRBUF+4                         ; 3314
    08  AE                         B2   D0 0005A     MOVL    (PTR)+, CTRBUF+8                         ; 3315
    0C  AE                    04   A2   D0 0005E     MOVL    4(PTR), CTRBUF+12                        ; 3316
    10  AE                    08   A2   3C 00063     MOVZWL  8(PTR), CTRBUF+16                        ; 3317
    14  AE                    0A   A2   9A 00068     MOVZBL  10(PTR), CTRBUF+20                       ; 3318
    18  AE                    0B   A2   9A 0006D     MOVZBL  11(PTR), CTRBUF+24                       ; 3319
                                 52            0C   C0 00072     ADDL2   #12, PTR                         ; 3321
                                               62   95 00075     TSTB    (PTR)                            ; 3323
                                               07   12 00077     BNEQ    4$                               ;
                                 6E      0000' CF   9E 00079     MOVAB   UNKNOWN, CTRBUF                  ; 3325
                                               10   11 0007E     BRB     5$                               ;
                                 6E        60  AE   9E 00080 4$: MOVAB   DSC, CTRBUF                      ; 3328
                                            62  AE   B4 00084     CLRW    DSC+2                            ; 3329
    60  AE                                  62   9B 00087     MOVZBW  (PTR), DSC                       ;
    64  AE                    01   A2   9E 0008B     MOVAB   1(R2), DSC+4                             ;
                                 66      0400  8F   3C 00090 5$: MOVZWL  #1024, OUTDSC                    ; 3333
    04  A6                    08   A6   9E 00095     MOVAB   OUTBUF, OUTDSC+4                         ;
                                       4040  8F   BB 0009A     PUSHR   #^M<R6,SP>                       ; 3341
```

```
                                        56  DD 0009E          PUSHL   R6                                  ;
                                0000'   CF  9F 000A0          PUSHAB  SERVCTRSTR                          ;
                00000000G  00           04  FB 000A4          CALLS   #4, SYS$FAOL                        ;
                           53           50  D0 000AB          MOVL    R0, STATUS                          ;
                           05           53  E8 000AE          BLBS    STATUS, 6$                          ; 3343
                                        53  DD 000B1          PUSHL   STATUS                              ;
                           67           01  FB 000B3          CALLS   #1, LIB$SIGNAL                      ;
                                        56  DD 000B6  6$:     PUSHL   R6                                  ; 3345
                00000000G  00           01  FB 000B8          CALLS   #1, LIB$PUT_OUTPUT                  ;
                           53           50  D0 000BF          MOVL    R0, STATUS                          ;
                           05           53  E8 000C2          BLBS    STATUS, 7$                          ; 3347
                                        53  DD 000C5          PUSHL   STATUS                              ;
                           67           01  FB 000C7          CALLS   #1, LIB$SIGNAL                      ;
                           54           21  C0 000CA  7$:     ADDL2   #33, INDX                           ;
                           55           54  D1 000CD  8$:     CMPL    INDX, R5                            ;
                                        80  15 000D0          BLEQ    3$                                  ;
                                            04 000D2          RET                                         ; 3354

; Routine Size:  211 bytes,    Routine Base:  $CODE$ + 0ED2
```

```
%SBTTL  'LCP_SHOWHIST  Display the history buffer'
ROUTINE LCP_SHOWHIST : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Obtain the history buffer from the driver and then display it.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       history buffer in the driver.
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       errors signalled
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        STATUS,
        CHAR
        ;


    !
    ! Get the history buffer if there is one.
    !
    IF NOT
        (STATUS = $CMKRNL ( ROUTIN = LCPK_GETHISTORY ) )
    THEN
        BEGIN
        SIGNAL (.STATUS);
        RETURN;
        END
    ;

    !
    ! Now display the history buffer.  First the title line and then
    ! the buffer a line at a time.
    !
    LCP_FAOOUTPUT (FAOSTR1, 0);
    INCR IDX FROM AREA TO AREA + GHB$K_HISTSIZE - 32 BY 32
```

```
; 3304        3412  2     DO
; 3305        3413  2         LCP_FAOOUTPUT (FAOSTR2, .IDX)
; 3306        3414  2     ;
; 3307        3415  2
; 3308        3416  1     END;


                                  000C 00000 LCP_SHOWHIST:
                                                     .WORD   Save R2,R3                          ; 3356
                                 7E  D4 00002        CLRL    -(SP)                               ; 3398
                          0000V  CF  9F 00004        PUSHAB  LCPK_GETHISTORY                     :
                 00000000G  00    02  FB 00008        CALLS   #2, SYS$CMKRNL                      :
                            0A    50  E8 0000F        BLBS    STATUS, 1$                          :
                                  50  DD 00012        PUSHL   STATUS                              ; 3401
                 00000000G  00    01  FB 00014        CALLS   #1, LIB$SIGNAL                      :
                                      04 0001B        RET                                         ; 3400
                                 7E  D4 0001C 1$:     CLRL    -(SP)                               ; 3410
                          0000'  CF  9F 0001E        PUSHAB  FAOSTR1                             :
                     0000V  CF    02  FB 00022        CALLS   #2, LCP_FAOOUTPUT                   :
                            52    0000' CF 9E 00027   MOVAB   AREA, R2                            ; 3411
                            53    0000' CF 9E 0002C   MOVAB   AREA+2528, R3                       :
                                  0E  11 00031        BRB     3$                                  :
                                  52  DD 00033 2$:     PUSHL   IDX                                 ; 3413
                          0000'  CF  9F 00035        PUSHAB  FAOSTR2                             :
                     0000V  CF    02  FB 00039        CALLS   #2, LCP_FAOOUTPUT                   :
                            52    20  C0 0003E        ADDL2   #32, IDX                            :
                            53    52  D1 00041 3$:     CMPL    IDX, R3                             :
                                  ED  15 00044        BLEQ    2$                                  :
                                      04 00046        RET                                         ; 3416

; Routine Size:  71 bytes,    Routine Base:  $CODE$ + 0FA5
```

```
%SBTTL  'LCP_SHOWCHAR  Display the LAT driver characteristics'
ROUTINE LCP_SHOWCHAR : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Obtain the common data area from the driver and then display
!   interesting tidbits.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       common data area in the driver.
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       errors signalled
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        STATUS,
        PTR,
        NUM_SERVICES
        ;

    LITERAL
        STRSIZE = 512,                  ! Size of input and output strings
        PRMSIZE = 24                    ! Size of parameter list in LONGs
        ;

    LOCAL
        DSC : BLOCK [DSC$K_Z_BLN, BYTE],
        BUF : VECTOR [STRSIZE, BYTE],
        STR : REF VECTOR [,BYTE],
        VER_ECO : VECTOR [2, BYTE],
        P : VECTOR [PRMSIZE]
        ;

    !
    !  Get the COMMON AREA in the driver and setup the descriptors.
    !
    IF NOT
```

```
3474  3          (STATUS = $CMKRNL ( ROUTIN = LCPK_GETCOMMON ) )
3475  2      THEN
3476  2          SIGNAL (.STATUS)
3477  2      ;
3478  2
3479  2      !
3480  2      !  Initialize descriptors
3481  2      !
3482  2      $SETDSC (IDDSC, GHB$K_IDLEN, ID);
3483  2
3484  2      $SETDSC (NODE_IDDSC, GHB$K_IDLEN, NODE_ID);
3485  2
3486  2      $SETDSC (SERVDSC, GHB$K_NAMELEN, SERV);
3487  2
3488  2      $SETDSC (NAMEDSC, GHB$K_NAMELEN, NAME);
3489  2
3490  2      STR = .OUTDSC [DSC$A_POINTER];
3491  2
3492  2      !
3493  2      !  Pick up group codes, timer and skip host node status
3494  2      !
3495  2      MCASTIMR = .STR [0];
3496  2      PTR = CH$MOVE ( CH$RCHAR (STR [2]), STR [3], GROUPS);
3497  2
3498  2      !
3499  2      !  Skip the group codes and host node name.
3500  2      !
3501  2      STR = .STR [2] + STR [3];            ! Skip group codes
3502  2      STR = .STR [0] + STR [1];            ! Skip host node name
3503  2
3504  2      !
3505  2      !  Get host node id, then skip over it.
3506  2      !
3507  2      NODE_IDDSC [DSC$W_LENGTH] = .STR [0];
3508  2      PTR = CH$MOVE (.NODE_IDDSC [DSC$W_LENGTH],
3509  2                     STR [1], .NODE_IDDSC [DSC$A_POINTER]);
3510  2
3511  2      STR = .STR [0] + STR [1];            ! Skip host node id.
3512  2
3513  2      !
3514  2      !  Get number of services
3515  2      !
3516  2      NUM_SERVICES = .STR [0];
3517  2
3518  2      !
3519  2      !  Get the 1st service name and descriptor
3520  2      !
3521  2      NAMEDSC [DSC$W_LENGTH] = .STR [2];
3522  2      PTR = CH$MOVE (.NAMEDSC [DSC$W_LENGTH], STR [3], .NAMEDSC [DSC$A_POINTER]);
3523  2
3524  2      STR = .STR [2] + STR [3];            ! Skip 1st service name
3525  2      IDDSC [DSC$W_LENGTH] = .STR [0];
3526  2      PTR = CH$MOVE (.IDDSC [DSC$W_LENGTH], STR [1], .IDDSC [DSC$A_POINTER]);
3527  2
3528  2      !
3529  2      ! If we have a second service name, then report the second service here
3530  2      !
```

```
: 3424          3531  2     IF .NUM_SERVICES  LEQ  1
: 3425          3532  2     THEN
: 3426          3533  2         SERVDSC [DSC$W_LENGTH] = 0
: 3427          3534  2     ELSE
: 3428          3535  3         BEGIN
: 3429          3536  3         STR = .STR [0] + STR [1];                   ! Skip 1st service id
: 3430          3537  3         STR = .STR + 1;                             ! Skip 2nd rating
: 3431          3538  3         SERVDSC [DSC$W_LENGTH] = .STR [0];
: 3432          3539  3         PTR = CH$MOVE (.SERVDSC [DSC$W_LENGTH], STR [1],
: 3433          3540  3                 .SERVDSC [DSC$A_POINTER])
: 3434          3541  3         END
: 3435          3542  2      ;
: 3436          3543  2
: 3437          3544  2     !
: 3438          3545  2     !  Now display the relevant parts of the common area.
: 3439          3546  2     !
: 3440          3547  2
: 3441          3548  2     !
: 3442          3549  2     !   Convert the group codes to a nice printable ascii string
: 3443          3550  2     !
: 3444          3551  2     LCP_CVTGROUPS ();
: 3445          3552  2
: 3446          3553  2     !
: 3447          3554  2     ! Set up the parameter list and report the current characteristics
: 3448          3555  2     !
: 3449          3556  2     P[0] = NAMEDSC;
: 3450          3557  2     P[1] = SERVDSC;
: 3451          3558  2     P[2] = NODE_IDDSC;
: 3452          3559  2     P[3] = IDDSC;
: 3453          3560  2     P[4] = GRPDSC;
: 3454          3561  2     P[5] = .MCASTIMR;
: 3455          3562  2     P[6] = .AREA [GHB$B_LATVERSION];
: 3456          3563  2     P[7] = .AREA [GHB$B_LATECO];
: 3457          3564  2
: 3458          3565  2     IF .AREA [GHB$L_TIM_ACT] NEQ 0
: 3459          3566  2     THEN
: 3460          3567  2         P[8] = ACTIVE
: 3461          3568  2     ELSE
: 3462          3569  2         P[8] = INACTIVE;
: 3463          3570  2
: 3464          3571  2     $SETDSC (DSC, STRSIZE, BUF);
: 3465          3572  2
: 3466        P 3573  2     $FAOL
: 3467        P 3574  2         (
: 3468        L 3575  2         CTRSTR = %ASCID
: 3469        L 3576  2         %STRING
: 3470        L 3577  2             (
: 3471        L 3578  2             '!/LCP Characteristics!/!/',
: 3472        L 3579  2             'Node name = !18<\!AS\!>   Service name = !18<\!AS\!>!/',
: 3473        L 3580  2             'Node Identification = \!AS\!/',
: 3474        L 3581  2             'Service Identification = \!AS\!/',
: 3475        L 3582  2             'Groups = !AS!/',
: 3476        L 3583  2             'Multicast timer = !UB seconds!/',
: 3477        L 3584  2             'LAT Version = !16<!UB.!UB!>   LAT Protocol is !AS!/'
: 3478        P 3585  2             ),
: 3479        P 3586  2         OUTBUF = DSC,
: 3480        P 3587  2         OUTLEN = DSC,
```

```
; 3481        P 3588  2          PRMLST = P[0]
; 3482          3589  2          );
; 3483          3590  2
; 3484          3591  2      LIB$PUT_OUTPUT (DSC)
; 3485          3592  2
; 3486          3593  1      END;


                                                                                  .PSECT  $PLIT$,NOWRT,NOEXE,2

72  65  74  63  61  72  61  68  43  20  50  43  4C  2F  21  00E20 P.AFT:  .ASCII  \!/LCP Characteristics!/!/Node name = !18\
20  65  64  6F  4E  2F  21  2F  21  73  63  69  74  73  69  00E2F
                    38  31  21  20  3D  20  65  6D  61  6E  00E3E
76  72  65  53  20  20  20  3E  21  5C  53  41  21  5C  3C  00E48         .ASCII  \<\<92>\!AS\<92>\!>   Service name = !1\
        31  21  20  3D  20  65  6D  61  6E  20  65  63  69  00E57
65  64  6F  4E  2F  21  3E  21  5C  53  41  21  5C  3C  38  00E64         .ASCII  \8<\<92>\!AS\<92>\!>!/Node Identificati\
        69  74  61  63  69  66  69  74  6E  65  64  49  20  00E73
72  65  53  2F  21  5C  53  41  21  5C  20  3D  20  6E  6F  00E80         .ASCII  \on = \<92>\!AS\<92>\!/Service Identifi\
        69  66  69  74  6E  65  64  49  20  65  63  69  76  00E8F
21  5C  53  41  21  5C  20  3D  20  6E  6F  69  74  61  63  00E9C         .ASCII  \cation = \<92>\!AS\<92>\!/Groups = !AS\
        53  41  21  20  3D  20  73  70  75  6F  72  47  2F  00EAB
6D  69  74  20  74  73  61  63  69  74  6C  75  4D  2F  21  00EB8         .ASCII  \!/Multicast timer = !UB seconds!/LAT Ver\
64  6E  6F  63  65  73  20  42  55  21  20  3D  20  72  65  00EC7
                    72  65  56  20  54  41  4C  2F  21  73  00ED6
2E  42  55  21  3C  36  31  21  20  3D  20  6E  6F  69  73  00EE0         .ASCII  \sion = !16<!UB.!UB!>   LAT Protocol is !\
6F  72  50  20  54  41  4C  20  20  20  3E  21  42  55  21  00EEF
                    21  20  73  69  20  6C  6F  63  6F  74  00EFE
                                            2F  21  53  41  00F08         .ASCII  \AS!/\
                                                    010E00EC  00F0C P.AFS:  .LONG   17694956
                                                    00000000'  00F10        .ADDRESS P.AFT


                                                                                  .PSECT  $CODE$,NOWRT,2

                                        03FC 00000 LCP_SHOWCHAR:
                                                                          .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                    ; 3418
                59      0000'   CF  9E 00002                              MOVAB   SERVDSC, R9                                     ;
                5E      FD98    CE  9E 00007                              MOVAB   -616(SP), SP                                    ;
                                7E  D4 0000C                              CLRL    -(SP)                                           ; 3474
                        0000V   CF  9F 0000E                              PUSHAB  LCPK_GETCOMMON                                  ;
00000000G       00              02  FB 00012                              CALLS   #2, SYS$CMKRNL                                  ;
                09              50  E8 00019                              BLBS    STATUS, 1$                                      ;
                                50  DD 0001C                              PUSHL   STATUS                                          ; 3476
00000000G       00              01  FB 0001E                              CALLS   #1, LIB$SIGNAL                                  ;
        48      A9      40      8F  9A 00025 1$:                          MOVZBL  #64, IDDSC                                      ; 3482
        4C      A9      08      A9  9E 0002A                              MOVAB   ID, IDDSC+4                                     ;
      0090      C9      40      8F  9A 0002F                              MOVZBL  #64, NODE_IDDSC                                 ; 3484
      0094      C9      50      A9  9E 00035                              MOVAB   NODE_ID, NODE_IDDSC+4                           ;
                69              10  D0 0003B                              MOVL    #16, SERVDSC                                    ; 3486
        04      A9      F0      A9  9E 0003E                              MOVAB   SERV, SERVDSC+4                                 ;
        E8      A9              10  D0 00043                              MOVL    #16, NAMEDSC                                    ; 3488
        EC      A9      D8      A9  9E 00047                              MOVAB   NAME, NAMEDSC+4                                 ;
                56      0AF4    C9  D0 0004C                              MOVL    OUTDSC+4, STR                                   ; 3490
      00E0      C9              66  9A 00051                              MOVZBL  (STR), MCASTIMR                                 ; 3495
                50      02      A6  9A 00056                              MOVZBL  2(STR), R0                                      ; 3496
```

```
                        57      03   A6  9E 0005A          MOVAB   3(STR), R7                                  ;
  00B0   C9             67           50  28 0005E          MOVC3   R0, (R7), GROUPS                          ;
                        56      02   A6  9A 00064          MOVZBL  2(STR), STR                               ; 3501
                        56           57  C0 00068          ADDL2   R7, STR                                   ;
                        50           66  9A 0006B          MOVZBL  (STR), R0                                 ; 3502
                        50           56  C0 0006E          ADDL2   STR, R0                                   ;
                        56      01   A0  9E 00071          MOVAB   1(R0), STR                                ;
                0090    C9           66  9B 00075          MOVZBW  (STR), NODE_IDDSC                         ; 3507
                        57      01   A6  9E 0007A          MOVAB   1(STR), R7                                ; 3509
  0094   D9             67    0090   C9  28 0007E          MOVC3   NODE_IDDSC, (R7), @NODE_IDDSC+4           ;
                        56           66  9A 00086          MOVZBL  (STR), STR                                ; 3511
                        56           57  C0 00089          ADDL2   R7, STR                                   ;
                        58           66  9A 0008C          MOVZBL  (STR), NUM_SERVICES                       ; 3516
                   E8   A9      02   A6  9B 0008F          MOVZBW  2(STR), NAMEDSC                           ; 3521
                        57      03   A6  9E 00094          MOVAB   3(STR), R7                                ; 3522
   EC   B9              67      E8   A9  28 00098          MOVC3   NAMEDSC, (R7), @NAMEDSC+4                 ;
                        56      02   A6  9A 0009E          MOVZBL  2(STR), STR                               ; 3524
                        56           57  C0 000A2          ADDL2   R7, STR                                   ;
                   48   A9           66  9B 000A5          MOVZBW  (STR), IDDSC                              ; 3525
                        57      01   A6  9E 000A9          MOVAB   1(STR), R7                                ; 3526
   4C   B9              67      48   A9  28 000AD          MOVC3   IDDSC, (R7), @IDDSC+4                     ;
                        01           58  D1 000B3          CMPL    NUM_SERVICES, #1                          ; 3531
                                     04  14 000B6          BGTR    2$                                        ;
                                     69  B4 000B8          CLRW    SERVDSC                                   ; 3533
                                     11  11 000BA          BRB     3$                                        ;
                        56           66  9A 000BC 2$:      MOVZBL  (STR), STR                                ; 3536
                        56           57  C0 000BF          ADDL2   R7, STR                                   ;
                                     56  D6 000C2          INCL    STR                                       ; 3537
                        69           66  9B 000C4          MOVZBW  (STR), SERVDSC                            ; 3538
   04   B9         01   A6           69  28 000C7          MOVC3   SERVDSC, 1(STR), @SERVDSC+4               ; 3540
                0000V   CF           00  FB 000CD 3$:      CALLS   #0, LCP_CVTGROUPS                         ; 3551
                        6E      E8   A9  9E 000D2          MOVAB   NAMEDSC, P                                ; 3556
                   04   AE           69  9E 000D6          MOVAB   SERVDSC, P+4                              ; 3557
                   08   AE    0090   C9  9E 000DA          MOVAB   NODE_IDDSC, P+8                           ; 3558
                   0C   AE      48   A9  9E 000E0          MOVAB   IDDSC, P+12                               ; 3559
                   10   AE    0FF8   C9  9E 000E5          MOVAB   GRPDSC, P+16                              ; 3560
                   14   AE    00E0   C9  D0 000EB          MOVL    MCASTIMR, P+20                            ; 3561
                   18   AE    00EE   C9  9A 000F1          MOVZBL  AREA+2, P+24                              ; 3562
                   1C   AE    00EF   C9  9A 000F7          MOVZBL  AREA+3, P+28                              ; 3563
                              0104   C9  D5 000FD          TSTL    AREA+24                                   ; 3565
                                     08  13 00101          BEQL    4$                                        ;
                   20   AE    0000'  CF  9E 00103          MOVAB   ACTIVE, P+32                              ; 3567
                                     06  11 00109          BRB     5$                                        ;
                   20   AE    0000'  CF  9E 0010B 4$:      MOVAB   INACTIVE, P+32                            ; 3569
                   F8   AD    0200   8F  3C 00111 5$:      MOVZWL  #512, DSC                                 ; 3571
                   FC   AD      60   AE  9E 00117          MOVAB   BUF, DSC+4                                ;
                                     5E  DD 0011C          PUSHL   SP                                        ; 3589
                                F8   AD  9F 0011E          PUSHAB  DSC                                       ;
                                F8   AD  9F 00121          PUSHAB  DSC                                       ;
                              0000'  CF  9F 00124          PUSHAB  P.AFS                                     ;
            00000000G   00           04  FB 00128          CALLS   #4, SYS$FAOL                              ;
                                F8   AD  9F 0012F          PUSHAB  DSC                                       ; 3591
            00000000G   00           01  FB 00132          CALLS   #1, LIB$PUT_OUTPUT                        ;
                                     04  00139             RET                                               ; 3593

; Routine Size:  314 bytes,    Routine Base:  $CODE$ + 0FEC
```

```
%SBTTL  'LCP_CVTGROUPS  Convert the group codes to an ascii printable string'
ROUTINE LCP_CVTGROUPS : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Covert the group codes in thr GROUPS vector to an ascii output
!   string, suitable for printing.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       GROUP vector area set up with group codes
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       errors signalled
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        STATUS,
        BITPOS,                         ! Bit position within longword
        SIZE,                           ! Length of bit stream to scan
        BYTE_OFFSET,                    ! Offset in group code string
        PTR,                            ! Pointer into output string
        LEFT_PAREN,
        RIGHT_PAREN,
        COMMA,
        INDEX
        ;

    LITERAL
        STRSIZE = 512,                  ! Size of input and output strings
        BITSIZE = 32,                   ! Maximum number of bits to scan
        CVTSTRSIZE = 3                  ! Size of output conversion string
        ;

    LOCAL
        CVTDSC : BLOCK [DSC$K_Z_BLN, BYTE],
        CVTSTR : VECTOR [CVTSTRSIZE, BYTE],
        GROUP : REF BLOCK [,BYTE]
        ;
```

```
; 3545          3651  2
; 3546          3652  2     LEFT_PAREN = %C'('; RIGHT_PAREN = %C')'; COMMA = %C',';
; 3547          3653  2
; 3548          3654  2     !
; 3549          3655  2     !  Initialize the group codes input string
; 3550          3656  2     !
; 3551          3657  2     $SETDSC (GRPDSC, STRSIZE, GRPBUF);
; 3552          3658  2
; 3553          3659  2     PTR = .GRPDSC [DSC$A_POINTER];
; 3554          3660  2
; 3555          3661  2     PTR = CH$MOVE(1, LEFT_PAREN, .PTR);
; 3556          3662  2
; 3557          3663  2     !
; 3558          3664  2     !  Initialize the conversion output string
; 3559          3665  2     !
; 3560          3666  2     $SETDSC (CVTDSC, CVTSTRSIZE, CVTSTR);
; 3561          3667  2
; 3562          3668  2     !
; 3563          3669  2     ! Set up the Group codes string
; 3564          3670  2     !
; 3565          3671  2
; 3566          3672  2     BYTE_OFFSET = 0;                    ! Start from beginning
; 3567          3673  2     GROUP = GROUPS;                     ! Copy the group code pointer
; 3568          3674  2     !
; 3569          3675  2     !   Sear the group codes one longword at a time, note that there are
; 3570          3676  2     !   32 longwords that we must search. We will use the FFS library routine
; 3571          3677  2     !   to find all corresponding bits that are set.
; 3572          3678  2     !
; 3573          3679  2     WHILE .BYTE_OFFSET LSS 32
; 3574          3680  2     DO
; 3575          3681  3         BEGIN
; 3576          3682  3         BITPOS = 0;
; 3577          3683  3         SIZE = BITSIZE;
; 3578          3684  3         STATUS = TRUE;
; 3579          3685  3         WHILE .STATUS AND .BITPOS LSS BITSIZE
; 3580          3686  3         DO
; 3581          3687  4             BEGIN
; 3582          3688  4             STATUS = LIB$FFS(BITPOS, SIZE, GROUP[.BYTE_OFFSET,0,0,0],
; 3583          3689  4                              BITPOS);
; 3584          3690  4             IF .STATUS
; 3585          3691  4             THEN
; 3586          3692  5                 BEGIN
; 3587          3693  5                 !
; 3588          3694  5                 !   Convert the bit number + offset to a binary value,
; 3589          3695  5                 !   and strip off leading spaces from converted string.
; 3590          3696  5                 !
; 3591          3697  5                 STATUS = OTS$CVT_L_TI (%REF((.BYTE_OFFSET*8)+.BITPOS), CVTDSC);
; 3592          3698  5                 INDEX = LIB$SKPC(%ASCID %STRING(' '), CVTDSC) - 1;
; 3593          3699  5                 !
; 3594          3700  5                 !   Move the converted string to the output buffer and
; 3595          3701  5                 !   add a "," to the end.
; 3596          3702  5                 !
; 3597          3703  5                 PTR = CH$MOVE (.CVTDSC [DSC$W_LENGTH] - .INDEX,
; 3598          3704  5                         .CVTDSC [DSC$A_POINTER] + .INDEX, .PTR);
; 3599          3705  5                 PTR = CH$MOVE(1, COMMA, .PTR);
; 3600          3706  5                 BITPOS = .BITPOS + 1;
; 3601          3707  5                 SIZE = .SIZE - .BITPOS
```

```
; 3602   3708  5                END
; 3603   3709  4            ;
; 3604   3710  4            END
; 3605   3711  3        ;
; 3606   3712  3        BYTE_OFFSET = .BYTE_OFFSET + 4
; 3607   3713  3        END
; 3608   3714  2    ;
; 3609   3715  2
; 3610   3716  2    !
; 3611   3717  2    !  Overwrite last comma with the right parenthesis
; 3612   3718  2    !
; 3613   3719  2    PTR = .PTR - 1;
; 3614   3720  2    PTR = CH$MOVE(1, RIGHT_PAREN, .PTR);
; 3615   3721  2
; 3616   3722  2    !
; 3617   3723  2    !  Calculate size of string moved
; 3618   3724  2    !
; 3619   3725  2    GRPDSC [DSC$W_LENGTH] = .PTR - .GRPDSC [DSC$A_POINTER];
; 3620   3726  2
; 3621   3727  1    END;


                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                        00  00  00  20  00F14 P.AFV:  .ASCII  \ \<0><0><0>                          ;
                                 010E0001  00F18 P.AFU:  .LONG   17694721                           ;
                                 00000000' 00F1C         .ADDRESS P.AFV                             ;


                                                    .PSECT  $CODE$,NOWRT,2

                                 0FFC 00000 LCP_CVTGROUPS:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11   ; 3595
                        5E            1C  C2 00002         SUBL2   #28, SP                        ;
                        50            28  D0 00005         MOVL    #40, LEFT_PAREN                ; 3652
                 04     AE            29  D0 00008         MOVL    #41, RIGHT_PAREN               ;
                        5B            2C  D0 0000C         MOVL    #44, COMMA                     ;
             0000'      CF     0200   8F  3C 0000F         MOVZWL  #512, GRPDSC                   ; 3657
             0000'      CF     0000'  CF  9E 00016         MOVAB   GRPBUF, GRPDSC+4               ;
                        53     0000'  CF  D0 0001D         MOVL    GRPDSC+4, PTR                  ; 3659
                        83            50  90 00022         MOVB    LEFT_PAREN, (PTR)+             ; 3661
                 14     AE            03  D0 00025         MOVL    #3, CVTDSC                     ; 3666
                 18     AE       08   AE  9E 00029         MOVAB   CVTSTR, CVTDSC+4               ;
                                      57  D4 0002E         CLRL    BYTE_OFFSET                    ; 3672
                        56     0000'  CF  9E 00030         MOVAB   GROUPS, GROUP                  ; 3673
                        20            57  D1 00035 1$:     CMPL    BYTE_OFFSET, #32               ; 3679
                                      73  18 00038         BGEQ    4$                             ;
                 0C     AE            20  7D 0003A         MOVQ    #32, SIZE                      ; 3683
                        5A            01  D0 0003E         MOVL    #1, STATUS                     ; 3684
             59         56            57  C1 00041         ADDL3   BYTE_OFFSET, GROUP, R9         ; 3688
                        60            5A  E9 00045 2$:     BLBC    STATUS, 3$                     ; 3685
                        20       10   AE  D1 00048         CMPL    BITPOS, #32                    ;
                                      5A  18 0004C         BGEQ    3$                             ;
                                 10   AE  9F 0004E         PUSHAB  BITPOS                         ; 3688
                                      59  DD 00051         PUSHL   R9                             ;
```

```
                                14   AE  9F 00053        PUSHAB  SIZE                                  ;
                                1C   AE  9F 00056        PUSHAB  BITPOS                                ;
            00000000G  00            04  FB 00059        CALLS   #4, LIB$FFS                           ;
                       5A            50  D0 00060        MOVL    R0, STATUS                            ;
                       42            5A  E9 00063        BLBC    STATUS, 3$                            ; 3690
                                14   AE  9F 00066        PUSHAB  CVTDSC                                ; 3697
                 04    AE       14 BE47  7E 00069        MOVAQ   @BITPOS[BYTE_OFFSET], 4(SP)           ;
                                04   AE  9F 0006F        PUSHAB  4(SP)                                 ;
            00000000G  00            02  FB 00072        CALLS   #2, OTS$CVT_L_TI                      ;
                       5A            50  D0 00079        MOVL    R0, STATUS                            ;
                                14   AE  9F 0007C        PUSHAB  CVTDSC                                ; 3698
                              0000'  CF  9F 0007F        PUSHAB  P.AFU                                 ;
            00000000G  00            02  FB 00083        CALLS   #2, LIB$SKPC                          ;
                       58       FF   A0  9E 0008A        MOVAB   -1(R0), INDEX                         ;
                       50       14   AE  3C 0008E        MOVZWL  CVTDSC, R0                            ; 3703
                       50            58  C2 00092        SUBL2   INDEX, R0                             ;
        63       18 BE48            50  28 00095        MOVC3   R0, @CVTDSC+4[INDEX], (PTR)           ; 3704
                       83            5B  90 0009B        MOVB    COMMA, (PTR)+                         ; 3705
                                10   AE  D6 0009E        INCL    BITPOS                                ; 3706
                 0C    AE       10   AE  C2 000A1        SUBL2   BITPOS, SIZE                          ; 3707
                                       9D  11 000A6        BRB     2$                                    ; 3685
                       57            04  C0 000A8 3$:    ADDL2   #4, BYTE_OFFSET                       ; 3712
                                     88  11 000AB        BRB     1$                                    ;
                       73       04   AE  90 000AD 4$:    MOVB    RIGHT_PAREN, -(PTR)                   ; 3720
                                     53  D6 000B1        INCL    PTR                                   ;
      0000'  CF        53     0000'  CF  A3 000B3        SUBW3   GRPDSC+4, PTR, GRPDSC                 ; 3725
                                     04 000BB        RET                                           ; 3727

; Routine Size:  188 bytes,    Routine Base:  $CODE$ + 1126
```

```
1 %SBTTL  'LCP_SHOWSERVERS  Display the LAT Servers'
1 ROUTINE LCP_SHOWSERVERS : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !   Show the LAT servers (by name) and number of active sessions.
1 !   interesting tidbits.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !        NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !        common data area in the driver.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !        NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !        errors signalled
1 !
1 ! SIDE EFFECTS:
1 !
1 !        NONE
1 !
1 !--
1
2     BEGIN
2
2     BIND
2         UNKNOWN = DESCRIPTOR ('UNKNOWN')
2         ;
2
2     LITERAL
2         BUFSIZE = 24
2         ;
2
2     LOCAL
2         NAME_DSC : BLOCK [DSC$K_Z_BLN, BYTE],
2         PARAM_BUF : VECTOR [BUFSIZE],
2         PTR : REF BLOCK [,BYTE],
2         STATUS
2         ;
2
2
2     CH$FILL (255, GHB$S_COMM_AREA, AREA);
2
2     !
2     !  Get the counters if we can, and if not then signal the error
2     !
2     IF NOT
3         (STATUS = $CMKRNL (ROUTIN = LCPK_GETSERVERS) )
```

```
; 3680    3785  2     THEN
; 3681    3786  2         SIGNAL (.STATUS)
; 3682    3787  2     ;
; 3683    3788  2
; 3684    3789  2     !
; 3685    3790  2     !  Convert the server info for output one at a time
; 3686    3791  2     !
; 3687    3792  2     !           Display:
; 3688    3793  2     !
; 3689    3794  2     !                   Server name
; 3690    3795  2     !                   Server state
; 3691    3796  2     !                   Number of users
; 3692    3797  2     !                   Ethernet address
; 3693    3798  2     !
; 3694    3799  2     IF .AREA_RET_SIZE EQL 0
; 3695    3800  2     THEN
; 3696    3801  2         SIGNAL (LCP$_NOSERVERS)
; 3697    3802  2     ELSE
; 3698    3803  3         BEGIN
; 3699    3804  3         INCR INDX FROM AREA
; 3700    3805  4                 TO AREA + .AREA_RET_SIZE - (GHB$K_NAMELEN + 9)
; 3701    3806  3                 BY GHB$K_NAMELEN + 9
; 3702    3807  3         DO
; 3703    3808  4             BEGIN
; 3704    3809  4             PTR = .INDX;
; 3705    3810  4
; 3706    3811  4             !
; 3707    3812  4             !  Get name descriptor
; 3708    3813  4             !
; 3709    3814  4             IF .PTR [0, 0, 8, 0] EQL 0
; 3710    3815  4             THEN
; 3711    3816  4                 PARAM_BUF [0] = UNKNOWN
; 3712    3817  4             ELSE
; 3713    3818  5                 BEGIN
; 3714    3819  5                 PARAM_BUF [0] = NAME_DSC;
; 3715    3820  6                 $SETDSC (NAME_DSC, .PTR [0, 0, 8, 0], PTR [1, 0, 8, 0])
; 3716    3821  5                 END
; 3717    3822  4             ;
; 3718    3823  4
; 3719    3824  4             !
; 3720    3825  4             !  Skip over server name
; 3721    3826  4             !
; 3722    3827  4             PTR = .PTR + GHB$K_NAMELEN + 1;
; 3723    3828  4
; 3724    3829  4             !
; 3725    3830  4             !  Get the Ethernet address, 6 bytes (1-6)
; 3726    3831  4             !
; 3727    3832  4             PARAM_BUF [1] = .PTR [1, 0, 8, 0];
; 3728    3833  4             PARAM_BUF [2] = .PTR [2, 0, 8, 0];
; 3729    3834  4             PARAM_BUF [3] = .PTR [3, 0, 8, 0];
; 3730    3835  4             PARAM_BUF [4] = .PTR [4, 0, 8, 0];
; 3731    3836  4             PARAM_BUF [5] = .PTR [5, 0, 8, 0];
; 3732    3837  4             PARAM_BUF [6] = .PTR [6, 0, 8, 0];
; 3733    3838  4
; 3734    3839  4             PARAM_BUF [8] = .PTR [0, 0, 8, 0];          ! Get the # of users
; 3735    3840  4             !
; 3736    3841  4             !  Display current state
```

```
; 3737          3842  4              !
; 3738          3843  4              IF .PTR [7, 0, 8, 0]
; 3739          3844  4              THEN
; 3740          3845  4                  PARAM_BUF [7] = ACTIVE
; 3741          3846  4              ELSE
; 3742          3847  4                  PARAM_BUF [7] = INACTIVE
; 3743          3848  4              ;
; 3744          3849  4
; 3745          3850  4              $SETDSC (OUTDSC, OUTBUFSIZE, OUTBUF);
; 3746          3851  4
; 3747        P 3852  4              STATUS = $FAOL
; 3748        P 3853  4                  (
; 3749        L 3854  4                  CTRSTR = %ASCID
; 3750        L 3855  4                  %STRING
; 3751        L 3856  4                      (
; 3752        L 3857  4                      '!/LCP Server Characteristics for !AS!/!/',
; 3753        L 3858  4                      'Ethernet address = !XB-!XB-!XB-!XB-!XB-!XB !/',
; 3754        L 3859  4                      'Server is !18<!AS!>    Active users = !18<'UB!>!/'
; 3755        P 3860  4                      ),
; 3756        P 3861  4                  OUTLEN = OUTDSC [DSC$W_LENGTH],
; 3757        P 3862  4                  OUTBUF = OUTDSC,
; 3758        P 3863  4                  PRMLST = PARAM_BUF [0]
; 3759          3864  4                  );
; 3760          3865  4
; 3761          3866  4              IF NOT .STATUS THEN SIGNAL (.STATUS);
; 3762          3867  4
; 3763          3868  4              STATUS = LIB$PUT_OUTPUT (OUTDSC);
; 3764          3869  4
; 3765          3870  4              IF NOT .STATUS THEN SIGNAL (.STATUS)
; 3766          3871  4
; 3767          3872  4              END
; 3768          3873  3          ;
; 3769          3874  3
; 3770          3875  3
; 3771          3876  3          END
; 3772          3877  3
; 3773          3878  1      END;


                                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                4E  57  4F  4E  4B  4E  55  00F20 P.AFX:  .ASCII  \UNKNOWN\                                  ;
                                                            00F27         .BLKB   1
                                                 00000007   00F28 P.AFW:  .LONG   7                                          ;
                                                 00000000'  00F2C         .ADDRESS P.AFX                                     ;
68  43  20  72  65  76  72  65  53  20  50  43  4C  2F  21  00F30 P.AFZ:  .ASCII  \!/LCP Server Characteristics for !AS!/!/\  ;
66  20  73  63  69  74  73  69  72  65  74  63  61  72  61  00F3F                                                            ;
                        2F  21  2F  21  53  41  21  20  72  6F  00F4E                                                            ;
73  65  72  64  64  61  20  74  65  6E  72  65  68  74  45  00F58         .ASCII  \Ethernet address = !XB-!XB-!XB-!XB-!XB-!\   ;
42  58  21  2D  42  58  21  2D  42  58  21  20  3D  20  73  00F67                                                            ;
                    21  2D  42  58  21  2D  42  58  21  2D  00F76                                                            ;
20  73  69  20  72  65  76  72  65  53  2F  21  20  42  58  00F80         .ASCII  \XB !/Server is !18<!AS!>    Active users\   ;
63  41  20  20  20  20  3E  21  53  41  21  3C  38  31  21  00F8F                                                            ;
                    73  72  65  73  75  20  65  76  69  74  00F9E                                                            ;
00  2F  21  3E  21  42  55  21  3C  38  31  21  20  3D  20  00FA8         .ASCII  \ = !18<!UB!>!/\<0><0>                       ;
                                                        00  00FB7                                                            ;
```

```
                                                 010E0086  00FB8 P.AFY:  .LONG   17694854                                  ;
                                                 00000000'  00FBC        .ADDRESS P.AFZ                                    ;

                                                                UNKNOWN=            P.AFW


                                                                        .PSECT  $CODE$,NOWRT,2

                                                        00FC 00000 LCP_SHOWSERVERS:
                                                                        .WORD   Save R2,R3,R4,R5,R6,R7                    ; 3729
                                 57 00000000G  00   9E 00002            MOVAB   LIB$SIGNAL, R7                            ;
                                 56     0000'  CF   9E 00009            MOVAB   OUTDSC, R6                                ;
                                 5E        98  AE   9E 0000E            MOVAB   -104(SP), SP                              ;
  0060   8F       FF   8F        6E            00   2C 00012            MOVC5   #0, (SP), #255, #96, AREA                 ; 3778
                                          F5FC  C6      0001A                                                            ;
                                                7E   D4 0001D            CLRL    -(SP)                                     ; 3784
                                          0000V  CF   9F 0001F            PUSHAB  LCPK_GETSERVERS                           ;
                          00000000G  00         02   FB 00023            CALLS   #2, SYS$CMKRNL                            ;
                                 53             50   D0 0002A            MOVL    R0, STATUS                                ;
                                 05             53   E8 0002D            BLBS    STATUS, 1$                                ;
                                                53   DD 00030            PUSHL   STATUS                                    ; 3786
                                 67             01   FB 00032            CALLS   #1, LIB$SIGNAL                            ;
                                 52         FC  A6   D0 00035 1$:        MOVL    AREA_RET_SIZE, R2                         ; 3799
                                                0A   12 00039            BNEQ    2$                                        ;
                                    00000000G  8F   DD 0003B            PUSHL   #LCP$_NOSERVERS                           ; 3801
                                 67             01   FB 00041            CALLS   #1, LIB$SIGNAL                            ;
                                                     04 00044            RET                                               ;
                                 54      F5FC   C6   9E 00045 2$:        MOVAB   AREA, R4                                  ; 3804
                                 55      F5E3 C642   9E 0004A            MOVAB   AREA-25[R2], R5                           ; 3805
                                              0092   31 00050            BRW     10$                                       ; 3806
                                 52             54   D0 00053 3$:        MOVL    INDX, PTR                                 ; 3809
                                                62   95 00056            TSTB    (PTR)                                     ; 3814
                                                07   12 00058            BNEQ    4$                                        ;
                                 6E      0000'  CF   9E 0005A            MOVAB   UNKNOWN, PARAM_BUF                        ; 3816
                                                10   11 0005F            BRB     5$                                        ;
                                 6E         60  AE   9E 00061 4$:        MOVAB   NAME_DSC, PARAM_BUF                       ; 3819
                                            62  AE   B4 00065            CLRW    NAME_DSC+2                                ; 3820
                            60   AE             62   9B 00068            MOVZBW  (PTR), NAME_DSC                           ;
                            64   AE             01   A2 9E 0006C         MOVAB   1(R2), NAME_DSC+4                         ;
                                 52             11   C0 00071 5$:        ADDL2   #17, PTR                                  ; 3827
                            04   AE             01   A2 9A 00074         MOVZBL  1(PTR), PARAM_BUF+4                       ; 3832
                            08   AE             02   A2 9A 00079         MOVZBL  2(PTR), PARAM_BUF+8                       ; 3833
                            0C   AE             03   A2 9A 0007E         MOVZBL  3(PTR), PARAM_BUF+12                      ; 3834
                            10   AE             04   A2 9A 00083         MOVZBL  4(PTR), PARAM_BUF+16                      ; 3835
                            14   AE             05   A2 9A 00088         MOVZBL  5(PTR), PARAM_BUF+20                      ; 3836
                            18   AE             06   A2 9A 0008D         MOVZBL  6(PTR), PARAM_BUF+24                      ; 3837
                            20   AE             62   9A 00092            MOVZBL  (PTR), PARAM_BUF+32                       ; 3839
                                 08             07   A2 E9 00096         BLBC    7(PTR), 6$                                ; 3843
                            1C   AE      0000'  CF   9E 0009A            MOVAB   ACTIVE, PARAM_BUF+28                      ; 3845
                                                06   11 000A0            BRB     7$                                        ;
                            1C   AE      0000'  CF   9E 000A2 6$:        MOVAB   INACTIVE, PARAM_BUF+28                    ; 3847
                                 66       0400  8F   3C 000A8 7$:        MOVZWL  #1024, OUTDSC                             ; 3850
                            04   A6         08  A6   9E 000AD            MOVAB   OUTBUF, OUTDSC+4                          ;
                                          4040  8F   BB 000B2            PUSHR   #^M<R6,SP>                                ; 3864
                                                56   DD 000B6            PUSHL   R6                                        ;
                                          0000'  CF   9F 000B8            PUSHAB  P.AFY                                     ;
                          00000000G  00         04   FB 000BC            CALLS   #4, SYS$FAOL                              ;
```

```
                                53          50  D0 000C3          MOVL    R0, STATUS                                  ;
                                05          53  E8 000C6          BLBS    STATUS, 8$                                  ; 3866
                                            53  DD 000C9          PUSHL   STATUS                                      ;
                                67          01  FB 000CB          CALLS   #1, LIB$SIGNAL                              ;
                                            56  DD 000CE 8$:      PUSHL   R6                                          ; 3868
                        00000000G   00      01  FB 000D0          CALLS   #1, LIB$PUT_OUTPUT                          ;
                                53          50  D0 000D7          MOVL    R0, STATUS                                  ;
                                05          53  E8 000DA          BLBS    STATUS, 9$                                  ; 3870
                                            53  DD 000DD          PUSHL   STATUS                                      ;
                                67          01  FB 000DF          CALLS   #1, LIB$SIGNAL                              ;
                                54          19  C0 000E2 9$:      ADDL2   #25, INDX                                   ;
                                55          54  D1 000E5 10$:     CMPL    INDX, R5                                    ;
                                            03  14 000E8          BGTR    11$                                         ;
                                          FF66  31 000EA          BRW     3$                                          ;
                                                04 000ED 11$:     RET                                                 ; 3878

; Routine Size:  238 bytes,    Routine Base:  $CODE$ + 11E2
```

```
%SBTTL  'LCP_HELP Give HELP'
GLOBAL ROUTINE LCP_HELP : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!   This routine is the LATCP help facility, and will display
!   useful and informative explanations of the LATCP facility.
!
! CALLING SEQUECE:
!    standard
!
! INPUT PARAMETERS:
!    none
!
! IMPLICIT INPUTS:
!    none
!
! OUTPUTS PARAMETERS:
!    none
!
! IMPLICIT OUTPUTS:
!
!    The help text will be printed on SYS$OUTPUT.
!
! ROUTINES CALLED:
!
!    LBR$OUTPUT_HELP
!
! ROUTINE VALUE:
!    none
!
! SIGNALS
!    none
!
! SIDE EFFECTS:
!    none
!
!--

    BEGIN

    EXTERNAL ROUTINE

        LBR$OUTPUT_HELP : ADDRESSING_MODE(GENERAL);

    BIND

        LIBRARY_NAME    = DESCRIPTOR ('LATCP'),         ! HELP text library
        KEYWORDS_DSC    = DESCRIPTOR ('KEYWORDS');      ! Keywords desc

    LOCAL

        SUBCMD_DSC      : BLOCK [DSC$K_Z_BLN, BYTE],    ! Rest of line
        STATUS;
```

```
; 3832          3936  2
; 3833          3937  2     Get help keywords and uppercase them
; 3834          3938  2
; 3835          3939  2      CH$FILL (0, DSC$C_Z_BLN, SUBCMD_DSC);                  ! Zero fill desc
; 3836          3940  2      SUBCMD_DSC [DSC$B_CLASS] = DSC$K_CLASS_D;   ! Fill in desc class
; 3837          3941  2      CLI$GET_VALUE (KEYWORDS_DSC, SUBCMD_DSC);   ! Get the KEYWORDS
; 3838          3942  2      IF .SUBCMD_DSC [DSC$W_LENGTH] NEQ 0
; 3839          3943  2      THEN
; 3840          3944  2          STR$UPCASE (SUBCMD_DSC, SUBCMD_DSC);
; 3841          3945  2
; 3842          3946  2
; 3843          3947  2 !   Call LBR$OUTPUT_HELP to do all the real work.
; 3844          3948  2 !
; 3845          3949  2      LBR$OUTPUT_HELP (LIB$PUT_OUTPUT, 0, SUBCMD_DSC,
; 3846          3950  2                   LIBRARY_NAME, 0, LIB$GET_INPUT);
; 3847          3951  1      END;


                                                                  .PSECT  $PLIT$,NOWRT,NOEXE,2

                        50  43  54  41  4C  00FC0 P.AGB:  .ASCII  \LATCP\                          ;
                                            00FC5         .BLKB   3
                                 00000005   00FC8 P.AGA:  .LONG   5                                ;
                                 00000000'  00FCC         .ADDRESS P.AGB                           ;
                    53  44  52  4F  57  59  45  4B  00FD0 P.AGD:  .ASCII  \KEYWORDS\                ;
                                 00000008   00FD8 P.AGC:  .LONG   8                                ;
                                 00000000'  00FDC         .ADDRESS P.AGD                           ;

                                                        LIBRARY_NAME=       P.AGA
                                                        KEYWORDS_DSC=       P.AGC
                                                                  .EXTRN  LBR$OUTPUT_HELP

                                                                  .PSECT  $CODE$,NOWRT,2

                                          003C 00000              .ENTRY  LCP_HELP, Save R2,R3,R4,R5   ; 3880
                        5E                08  C2 00002             SUBL2   #8, SP                       ;
08              00      6E                00  2C 00005             MOVC5   #0, (SP), #0, #8, SUBCMD_DSC ; 3939
                                          6E     0000A                                                 ;
                  03    AE                02  90 0000B             MOVB    #2, SUBCMD_DSC+3             ; 3940
                                          5E  DD 0000F             PUSHL   SP                           ; 3941
                            0000'         CF  9F 00011             PUSHAB  KEYWORDS_DSC                 ;
            000000C0G   00                02  FB 00015             CALLS   #2, CLI$GET_VALUE            ;
                                          6E  B5 0001C             TSTW    SUBCMD_DSC                   ; 3942
                                          0C  13 0001E             BEQL    1$                           ;
                                          5E  DD 00020             PUSHL   SP                           ; 3944
                              04          AE  9F 00022             PUSHAB  SUBCMD_DSC                   ;
            00000000G   00                02  FB 00025             CALLS   #2, STR$UPCASE               ;
                        00000000G         00  9F 0002C 1$:         PUSHAB  LIB$GET_INPUT                ; 3949
                                          7E  D4 00032             CLRL    -(SP)                        ;
                            0000'         CF  9F 00034             PUSHAB  LIBRARY_NAME                 ;
                              0C          AE  9F 00038             PUSHAB  SUBCMD_DSC                   ;
                                          7E  D4 0003B             CLRL    -(SP)                        ;
                        00000000G         00  9F 0003D             PUSHAB  LIB$PUT_OUTPUT               ;
            00000000G   00                06  FB 00043             CALLS   #6, LBR$OUTPUT_HELP          ;
                                              04 0004A             RET                                  ; 3951
```

```
; Routine Size:  75 bytes,    Routine Base:  $CODE$ + 12D0

; 3848          3952  1
```

```
%SBTTL  'LCP_FAOOUTPUT  Convert and write a string'
ROUTINE LCP_FAOOUTPUT (FAOSTR, DATA) : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Convert 32 bytes to a suitable parameter list for fao.  The fao
!   parameter string and the address of the data are passed.
!
! FORMAL PARAMETERS:
!
!       FAOSTR          Address of desc of fao control string
!       DATA            Address of 8 bytes of data
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    MAP
        DATA : REF VECTOR [, BYTE]
        ;

    LOCAL
        STATUS,
        EXPLODE : VECTOR [32, LONG]
        ;

    !
    ! Set the output buffer descriptor
    !
    $SETDSC (OUTDSC, OUTBUFSIZE, OUTBUF);

    !
    ! Explode the data to longwords eight bytes of a time.
    !
    IF .DATA NEQ 0
    THEN
        BEGIN
        EXPLODE [ 0] = DATA [0];        ! Get address of time
        INCR IDX FROM 8 TO 31
```

```
; 3907          4010  3          DO
; 3908          4011  4              BEGIN
; 3909          4012  4              EXPLODE [.IDX-7] = .DATA [.IDX]
; 3910          4013  4              END
; 3911          4014  3          END
; 3912          4015  2      ;
; 3913          4016  2
; 3914          4017  2      !
; 3915          4018  2      ! Convert and print an output line
; 3916          4019  2      !
; 3917        P 4020  2      STATUS = $FAOL
; 3918        P 4021  2          (
; 3919        P 4022  2          CTRSTR = .FAOSTR,
; 3920        P 4023  2          OUTLEN = OUTDSC [DSC$W_LENGTH],
; 3921        P 4024  2          OUTBUF = OUTDSC,
; 3922        P 4025  2          PRMLST = EXPLODE
; 3923          4026  2          );
; 3924          4027  2
; 3925          4028  2      IF NOT .STATUS THEN SIGNAL (.STATUS);
; 3926          4029  2
; 3927          4030  2      STATUS = LIB$PUT_OUTPUT (OUTDSC);
; 3928          4031  2
; 3929          4032  2      IF NOT .STATUS THEN SIGNAL (.STATUS)
; 3930          4033  2
; 3931          4034  1      END;


                          001C 00000 LCP_FAOOUTPUT:
                                          .WORD   Save R2,R3,R4                          ; 3954
            54 00000000G  00  9E 00002    MOVAB   LIB$_IGNAL, R4                         ;
            53     0000'  CF  9E 00009    MOVAB   OUTDSC, R3                             ;
            5E        80  AE  9E 0000E    MOVAB   -128(SP), SP                           ;
            63      0400  8F  3C 00012    MOVZWL  #1024, OUTDSC                          ; 4000
      04    A3        08  A3  9E 00017    MOVAB   OUTBUF, OUTDSC+4                       ;
            51        08  AC  D0 0001C    MOVL    DATA, R1                               ; 4005
                          10  13 00020    BEQL    2$                                     ;
            6E            51  D0 00022    MOVL    R1, EXPLODE                            ; 4008
            50            08  D0 00025    MOVL    #8, IDX                                ; 4009
      E4 AE40           6041  9A 00028 1$:MOVZBL  (IDX)[R1], EXPLODE-28[IDX]             ; 4012
F6          50            1F  F3 0002E    AOBLEQ  #31, IDX, 1$                           ; 4011
                        4008  8F  BB 00032 2$:PUSHR #^M<R3,SP>                           ; 4026
                          53  DD 00036    PUSHL   R3                                     ;
                          04  AC  DD 00038PUSHL   FAOSTR                                 ;
   00000000G  00            04  FB 0003B    CALLS   #4, SYS$FAOL                           ;
            52            50  D0 00042    MOVL    R0, STATUS                             ;
            05            52  E8 00045    BLBS    STATUS, 3$                             ; 4028
                          52  DD 00048    PUSHL   STATUS                                 ;
            64            01  FB 0004A    CALLS   #1, LIB$SIGNAL                         ;
                          53  DD 0004D 3$:PUSHL   R3                                     ; 4030
   00000000G  00            01  FB 0004F    CALLS   #1, LIB$PUT_OUTPUT                     ;
            52            50  D0 00056    MOVL    R0, STATUS                             ;
            05            52  E8 00059    BLBS    STATUS, 4$                             ; 4032
                          52  DD 0005C    PUSHL   STATUS                                 ;
            64            01  FB 0005E    CALLS   #1, LIB$SIGNAL                         ;
```

```
                                    04 00061 4$:     RET                                        ; 4034

; Routine Size:  98 bytes,    Routine Base:  $CODE$ + 131B
```

```
; 3933          4035  1 %SBTTL  'LCPK_GETCOUNTERS  Obtain the counterblock'
; 3934          4036  1 ROUTINE LCPK_GETCOUNTERS =
; 3935          4037  1
; 3936          4038  1 !++
; 3937          4039  1 ! FUNCTIONAL DESCRIPTION:
; 3938          4040  1 !
; 3939          4041  1 !   Copy the counters from the driver, to an area here for conversion.
; 3940          4042  1 !
; 3941          4043  1 ! FORMAL PARAMETERS:
; 3942          4044  1 !
; 3943          4045  1 !       NONE
; 3944          4046  1 !
; 3945          4047  1 ! IMPLICIT INPUTS:
; 3946          4048  1 !
; 3947          4049  1 !       counter block in driver
; 3948          4050  1 !
; 3949          4051  1 ! IMPLICIT OUTPUTS:
; 3950          4052  1 !
; 3951          4053  1 !       counter block here
; 3952          4054  1 !
; 3953          4055  1 ! ROUTINE VALUE:
; 3954          4056  1 ! COMPLETION CODES:
; 3955          4057  1 !
; 3956          4058  1 !       success or failure if no driver
; 3957          4059  1 !
; 3958          4060  1 ! SIDE EFFECTS:
; 3959          4061  1 !
; 3960          4062  1 !       NONE
; 3961          4063  1 !
; 3962          4064  1 !--
; 3963          4065  1
; 3964          4066  2     BEGIN
; 3965          4067  2
; 3966          4068  2     CH$MOVE (GHB$K_LENGTH, .LCP$L_AREA, AREA);
; 3967          4069  2     SUCCESS
; 3968          4070  2
; 3969          4071  1     END;


                                        003C 00000 LCPK_GETCOUNTERS:
                                                        .WORD   Save R2,R3,R4,R5                        ; 4036
        0000'  CF     0000'  DF     0060   8F  28 00002       MOVC3   #96, @LCP$L_AREA, AREA          ; 4068
                             50            01  D0 0000C       MOVL    #1, R0                          ; 4071
                                               04 0000F       RET                                     ;

; Routine Size:  16 bytes,    Routine Base:  $CODE$ + 137D
```

```
1 %SBTTL  'LCPK_GETSERVCTRS  Obtain the SERVER counters'
1 ROUTINE LCPK_GETSERVCTRS =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !   Copy the counters from the driver, to an area here for conversion.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       counter block in driver
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       counter block here
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       success or failure if no driver
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     LOCAL
2         CSB_LIST : REF VECTOR [, LONG],
2         CSB_PTR : REF VECTOR [, BYTE],
2         CSB_HEADER,
2         PTR
2         ;
2
2     !
2     !  Point to output buffer
2     !
2     PTR = AREA;
2
2     !
2     !  Get pointer to CSB table
2     !
2     CSB_LIST = .LCP$L_AREA [GHB$L_CSBLST];
2
2     !
2     !  Get the counters from each active CSB
2     !
2     IF .CSB_LIST NEQ 0
2     THEN
2         INCR INDX FROM 0 TO GHB$K_ACT_CSBS - 1
2         DO
```

```
; 4028          4129  3             BEGIN
; 4029          4130  3             CSB_PTR = .CSB_LIST [.INDX];
; 4030          4131  3             IF .CSB_PTR NEQ 0
; 4031          4132  3             THEN
; 4032          4133  3                 PTR = CH$MOVE (LCB_C_LENGTH + GHB$K_NAMELEN + 1,
; 4033          4134  3                                CSB_PTR [GHB$T_CSBCTR], .PTR)
; 4034          4135  3             END
; 4035          4136  2         ;
; 4036          4137  2
; 4037          4138  2     !
; 4038          4139  2     !  Get pointer to inactive CSB queue, go through queue in reverse order.
; 4039          4140  2     !
; 4040          4141  2     CSB_HEADER = LCP$L_AREA [GHB$Q_OLD_CSBS];
; 4041          4142  2
; 4042          4143  2     !
; 4043          4144  2     !  Get the counters from each inactive CSB
; 4044          4145  2     !
; 4045          4146  2     IF ..CSB_HEADER NEQ 0
; 4046          4147  2     THEN
; 4047          4148  3         BEGIN
; 4048          4149  3         CSB_PTR = .(.CSB_HEADER+4);
; 4049          4150  3         WHILE .CSB_PTR NEQ .CSB_HEADER
; 4050          4151  3         DO
; 4051          4152  4             BEGIN
; 4052          4153  4             PTR = CH$MOVE (LCB_C_LENGTH + GHB$K_NAMELEN + 1,
; 4053          4154  4                            CSB_PTR [GHB$T_CSBCTR], .PTR);
; 4054          4155  5             CSB_PTR = .(.CSB_PTR+4)
; 4055          4156  4             END
; 4056          4157  3         ;
; 4057          4158  3         END
; 4058          4159  2     ;
; 4059          4160  2     !
; 4060          4161  2     !  Calculate size of area filled in
; 4061          4162  2     !
; 4062          4163  2     AREA_RET_SIZE = .PTR - AREA;
; 4063          4164  2
; 4064          4165  2     SUCCESS
; 4065          4166  2
; 4066          4167  1     END;
```

```
                                    03FC 00000 LCPK_GETSERVCTRS:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9        ; 4073
                53      0000'  CF  9E 00002             MOVAB   AREA, PTR                           ; 4115
                56      0000'  CF  D0 00007             MOVL    LCP$L_AREA, R6                      ; 4120
                59        20   A6  D0 0000C             MOVL    32(R6), CSB_LIST                    ;
                                11  13 00010             BEQL    3$                                  ; 4125
                                58  D4 00012             CLRL    INDX                                ; 4127
                57            6948  D0 00014 1$:         MOVL    (CSB_LIST)[INDX], CSB_PTR           ; 4130
                                05  13 00018             BEQL    2$                                  ; 4131
63        0C    A7              21  28 0001A             MOVC3   #33, 12(CSB_PTR), (PTR)             ; 4134
F1              58              1F  F3 0001F 2$:         AOBLEQ  #31, INDX, 1$                       ; 4131
                56              24  C0 00023 3$:         ADDL2   #36, CSB_HEADER                     ; 4141
                                66  D5 00026             TSTL    (CSB_HEADER)                        ; 4146
```

```
                                    14  13 00028          BEQL    5$                                          ;
                        57     04   A6  D0 0002A          MOVL    4(CSB_HEADER), CSB_PTR                      ; 4149
                        56          57  D1 0002E 4$:      CMPL    CSB_PTR, CSB_HEADER                         ; 4150
                                    0B  13 00031          BEQL    5$                                          ;
                63   0C  A7          21  28 00033          MOVC3   #33, 12(CSB_PTR), (PTR)                     ; 4154
                        57     04   A7  D0 00038          MOVL    4(CSB_PTR), CSB_PTR                         ; 4155
                                    F0  11 0003C          BRB     4$                                          ;
                        50   0000'  CF  9E 0003E 5$:      MOVAB   AREA, R0                                    ; 4163
            0000'   CF  53          50  C3 00043          SUBL3   R0, PTR, AREA_RET_SIZE                      ;
                        50          01  D0 00049          MOVL    #1, R0                                      ; 4167
                                        04 0004C          RET                                                 ;

; Routine Size:  77 bytes,    Routine Base:  $CODE$ + 138D
```

```
%SBTTL  'LCPK_GETSERVERS  Obtain the SERVER information'
ROUTINE LCPK_GETSERVERS =

!++
! FUNCTIONAL DESCRIPTION:
!
!   Copy the CSB server info from the driver, to an area here for conversion.
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       CSB server block in driver
!
! IMPLICIT OUTPUTS:
!
!       CSB server block info here
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       success or failure if no driver
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        CSB_LIST : REF VECTOR [, LONG],
        CSB_PTR : REF VECTOR [, BYTE],
        CSB_HEADER,
        PTR,
        ACTIVE_V,
        INACTIVE_V
        ;

    ACTIVE_V = 1; INACTIVE_V = 0;
    !
    !  Point to output buffer
    !
    PTR = AREA;

    !
    !  Get pointer to CSB table
    !
    CSB_LIST = .LCP$L_AREA [GHB$L_CSBLST];

    !
    !  Get the info from each active CSB
    !
    IF .CSB_LIST NEQ 0
```

```
; 4125    4225  2     THEN
; 4126    4226  2         INCR INDX FROM 0 TO GHB$K_ACT_CSBS - 1
; 4127    4227  2         DO
; 4128    4228  3             BEGIN
; 4129    4229  3             CSB_PTR = .CSB_LIST [.INDX];
; 4130    4230  3             IF .CSB_PTR NEQ 0
; 4131    4231  3             THEN
; 4132    4232  4                 BEGIN
; 4133    4233  4                 PTR = CH$MOVE (GHB$K_NAMELEN + 8,
; 4134    4234  4                                CSB_PTR [GHB$T_CSBCTR + LCB_C_LENGTH], .PTR);
; 4135    4235  4                 PTR = CH$MOVE (1, ACTIVE_V, .PTR)
; 4136    4236  4                 END
; 4137    4237  3             ;
; 4138    4238  3             END
; 4139    4239  2         ;
; 4140    4240  2
; 4141    4241  2     !
; 4142    4242  2     !  Get pointer to inactive CSB queue, go through queue in reverse order.
; 4143    4243  2     !
; 4144    4244  2     CSB_HEADER = LCP$L_AREA [GHB$Q_OLD_CSBS];
; 4145    4245  2
; 4146    4246  2     !
; 4147    4247  2     !  Get the info from each inactive CSB
; 4148    4248  2     !
; 4149    4249  2     IF ..CSB_HEADER NEQ 0
; 4150    4250  2     THEN
; 4151    4251  3         BEGIN
; 4152    4252  3         CSB_PTR = .(.CSB_HEADER+4);
; 4153    4253  3         WHILE .CSB_PTR NEQ .CSB_HEADER
; 4154    4254  3         DO
; 4155    4255  4             BEGIN
; 4156    4256  4             PTR = CH$MOVE (GHB$K_NAMELEN + 8,
; 4157    4257  4                            CSB_PTR [GHB$T_CSBCTR + LCB_C_LENGTH], .PTR);
; 4158    4258  4             PTR = CH$MOVE (1, INACTIVE_V, .PTR);
; 4159    4259  5             CSB_PTR = .(.CSB_PTR+4)
; 4160    4260  4             END
; 4161    4261  3         ;
; 4162    4262  3         END
; 4163    4263  2     ;
; 4164    4264  2
; 4165    4265  2     !
; 4166    4266  2     !  Calculate size of area filled in
; 4167    4267  2     !
; 4168    4268  2     AREA_RET_SIZE = .PTR - AREA;
; 4169    4269  2
; 4170    4270  2     SUCCESS
; 4171    4271  2
; 4172    4272  1     END;


                                    0FFC 00000 LCPK_GETSERVERS:
                                                   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 4169
                 5A                 01  7D 00002   MOVQ    #1, ACTIVE_V                                ; 4210
                 53        0000'    CF  9E 00005   MOVAB   AREA, PTR                                   ; 4214
```

```
                        56    0000'  CF  D0 0000A        MOVL    LCP$L_AREA, R6                          ; 4219
                        59      20   A6  D0 0000F        MOVL    32(R6), CSB_LIST                        ;
                                     14  13 00013        BEQL    3$                                      ; 4224
                                     58  D4 00015        CLRL    INDX                                    ; 4226
                        57          6948  D0 00017 1$:   MOVL    (CSB_LIST)[INDX], CSB_PTR               ; 4229
                                     08  13 0001B        BEQL    2$                                      ; 4230
                63   1C  A7          18  28 0001D        MOVC3   #24, 28(CSB_PTR), (PTR)                 ; 4234
                        83          5A  90 00022        MOVB    ACTIVE_V, (PTR)+                        ; 4235
                EE      58          1F  F3 00025 2$:    AOBLEQ  #31, INDX, 1$                           ; 4226
                        56          24  C0 00029 3$:    ADDL2   #36, CSB_HEADER                         ; 4244
                                    66  D5 0002C        TSTL    (CSB_HEADER)                            ; 4249
                                    17  13 0002E        BEQL    5$                                      ;
                        57      04  A6  D0 00030        MOVL    4(CSB_HEADER), CSB_PTR                  ; 4252
                        56          57  D1 00034 4$:    CMPL    CSB_PTR, CSB_HEADER                     ; 4253
                                    0E  13 00037        BEQL    5$                                      ;
                63   1C  A7          18  28 00039        MOVC3   #24, 28(CSB_PTR), (PTR)                 ; 4257
                        83          5B  90 0003E        MOVB    INACTIVE_V, (PTR)+                      ; 4258
                        57      04  A7  D0 00041        MOVL    4(CSB_PTR), CSB_PTR                     ; 4259
                                    ED  11 00045        BRB     4$                                      ;
                        50    0000'  CF  9E 00047 5$:    MOVAB   AREA, R0                                ; 4268
            0000'   CF  53          50  C3 0004C        SUBL3   R0, PTR, AREA_RET_SIZE                  ;
                        50          01  D0 00052        MOVL    #1, R0                                  ; 4272
                                        04 00055        RET                                             ;

; Routine Size:  86 bytes,    Routine Base:  $CODE$ + 13DA
```

```
; 4174   1 %SBTTL  'LCPK_GETHISTORY  Obtain the history buffer'
; 4175   1 ROUTINE LCPK_GETHISTORY =
; 4176   1
; 4177   1 !++
; 4178   1 ! FUNCTIONAL DESCRIPTION:
; 4179   1 !
; 4180   1 !   Copy the history buffer from the driver, to an area here for conversion.
; 4181   1 !
; 4182   1 ! FORMAL PARAMETERS:
; 4183   1 !
; 4184   1 !       NONE
; 4185   1 !
; 4186   1 ! IMPLICIT INPUTS:
; 4187   1 !
; 4188   1 !       counter block in driver
; 4189   1 !
; 4190   1 ! IMPLICIT OUTPUTS:
; 4191   1 !
; 4192   1 !       counter block here
; 4193   1 !
; 4194   1 ! ROUTINE VALUE:
; 4195   1 ! COMPLETION CODES:
; 4196   1 !
; 4197   1 !       NONE
; 4198   1 !
; 4199   1 ! SIDE EFFECTS:
; 4200   1 !
; 4201   1 !       NONE
; 4202   1 !
; 4203   1 !--
; 4204   1
; 4205   2     BEGIN
; 4206   2
; 4207   2     LOCAL
; 4208   2         HIST_BUF : REF BLOCK [,BYTE]
; 4209   2         ;
; 4210   2
; 4211   2     !
; 4212   2     !  If no history buffer, return error
; 4213   2     !
; 4214   2     IF .LCP$L_AREA [GHB$L_HISTORY] EQL 0
; 4215   2     THEN
; 4216   2         RETURN LCP$_NOHISTORY
; 4217   2     ;
; 4218   2
; 4219   2     !
; 4220   2     !  Copy the history buffer
; 4221   2     !
; 4222   2     HIST_BUF = .LCP$L_AREA [GHB$L_HISTORY];
; 4223   2     CH$MOVE (GHB$K_HISTSIZE,
; 4224   2         HIST_BUF [RBF_Z_DATA], AREA);
; 4225   2
; 4226   2     SUCCESS
; 4227   2
; 4228   1     END;
```

```
                                        003C 00000 LCPK_GETHISTORY:
                                                        .WORD   Save R2,R3,R4,R5                        ; 4274
                        50      0000'  CF  D0 00002     MOVL    LCP$L_AREA, R0                          ; 4313
                                 08    A0  D5 00007     TSTL    8(R0)                                   ;
                                       08  12 0000A     BNEQ    1$                                      ;
                        50 00000000G   8F  D0 0000C     MOVL    #LCP$_NOHISTORY, R0                     ; 4315
                                           04 00013     RET                                             ;
                        50       08    A0  D0 00014 1$: MOVL    8(R0), HIST_BUF                         ; 4321
          0000'  CF      0C    A0     GA00  8F  28 00018  MOVC3   #2560, 12(HIST_BUF), AREA             ; 4323
                        50             01  D0 00021     MOVL    #1, R0                                  ; 4327
                                           04 00024     RET                                             ;

; Routine Size:  37 bytes,    Routine Base:  $CODE$ + 1430
```

```
1 %SBTTL  'LCPK_GETCOMMON  Kernel mode routine to get common buffer area'
1 ROUTINE LCPK_GETCOMMON =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !   Kernel mode routine to get the information from the common area in
1 !   driver.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !        NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !        results of the qualifier parse
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !        NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !        NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !        qualifier results set for multicast
1 !
1 !--
1
2     BEGIN
2
2     LOCAL
2         LT : REF BLOCK [,BYTE],
2         PTR,
2         STR : REF BLOCK [,BYTE]
2         ;
2
2     !
2     !  Setup the output descriptor
2     !
2     $SETDSC (OUTDSC, OUTBUFSIZE, OUTBUF);
2
2     !
2     ! Copy the common area
2     !
2     PTR = CH$MOVE (GHB$S_COMM_AREA, .LCP$L_AREA, AREA);
2
2     !
2     ! Copy the node information to the output buffer area
2     !
2     LT = .LCP$L_AREA;
2     STR = .LT [GHB$L_NODE];
2     PTR = .OUTDSC [DSC$A_POINTER];
2     PTR = CH$MOVE (.STR [-2, 0, 16, 0], .LT [GHB$L_NODE], .PTR);
```

```
; 4287          4385  2
; 4288          4386  2     SUCCESS
; 4289          4387  2
; 4290          4388  1     END;


                                                  007C 00000 LCPK_GETCOMMON:
                                                                .WORD   Save R2,R3,R4,R5,R6                     ; 4329
                                 56     0000'  CF  9E 00002    MOVAB   OUTDSC+4, R6                            ;
                              FC A6     0400   8F  3C 00007    MOVZWL  #1024, OUTDSC                           ; 4371
                                 66       04   A6  9E 0000D    MOVAB   OUTBUF, OUTDSC+4                        ;
                F5F8   C6     F5F4 D6     0060   8F  28 00011    MOVC3   #96, @LCP$L_AREA, AREA                  ; 4376
                                 50     F5F4   C6  D0 0001B    MOVL    LCP$L_AREA, LT                          ; 4381
                                 51       1C   A0  D0 00020    MOVL    28(LT), STR                             ; 4382
                                 53            66  D0 00024    MOVL    OUTDSC+4, PTR                           ; 4383
                       63     1C B0       FE   A1  28 00027    MOVC3   -2(STR), @28(LT), (PTR)                 ; 4384
                                 50            01  D0 0002D    MOVL    #1, R0                                  ; 4388
                                                   04 00030    RET                                             ;

; Routine Size:  49 bytes,    Routine Base:  $CODE$ + 1455
```

```
; 4292          4389  1 END                                .End of module
; 4293          4390  0 ELUDOM


                                                                    .EXTRN  LIB$SIGNAL

;                                   PSECT SUMMARY
;
;       Name                        Bytes                           Attributes
;
;  $PLIT$                            4064  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                             5968  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                            5254  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                                 Library Statistics
;
;                                          -------- Symbols --------      Pages       Processing
;       File                               Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1           18619        48         0       1000         00:01.8
;  _$255$DUA28:[SHRLIB]NMALIBRY.L32;1        887        35         3         47         00:01.4


;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:LATCP/OBJ=OBJ$:LATCP MSRC$:LATCP/UPDATE=(ENH$:LATCP)

; Size:          5254 code + 10032 data bytes
; Run Time:        01:52.9
; Elapsed Time:    03:55.0
; Lines/CPU Min:    2332
; Lexemes/CPU-Min: 20161
; Memory Used:  304 pages
; Compilation Complete
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

LAT

LATCPCLD
CLD

LATCP
MAP

LATDEF
SDL

LATCPMSG
LIS

LATCP
LIS

LTDRIVER
MAP

LTDRIVER
LIS